BENG (HONS) ELECTRICAL AND ELECTRONIC ENGINEERING

EN0624 Individual Engineering Project

PROJECT FINAL REPORT

2015/2016

PROJECT TITLE:

Control and Self-Balancing of Two-wheel Robots

Student Name: YEQIU TANG

Student ID: w14029808

Module code: EN0624

Supervisor: Dr Xuewu Dai

Submission date: 13/04/2016

Declaration

I hereby declare that the work contained in this dissertation has not been submitted for any other award and that it is all my own work. I also confirm that this work uses ideas and opinions from written papers and articles from the work of others.

Name: YEQIU TANG

Signature: _____

Date: _____

Abstract

With the rapid development of automobile industry, the quantity of automobile per capita raises steadily. In order to provide more flexibity and reduce carbon emittion, the concept of self-balancing two-wheel electrical vehicle emerged. It is a kind of clean and small urban electrical auxiliary vehicle which can solve the problem caused by traditional automobile.

This project aims to develop a self-balancing two-wheel robot system. The self-balancing two-wheel robot (SBTWR) is a nonlinear, strong coupling, underactuated and unstable system. The relevant results on the control method of which could be applied to other similar systems, it is of great theoretical significance.

Firstly, based on the combination of analytic mechanics and classical mechanics, the mathematical model of the system are built. The correctness of the models are validated in MATLAB® and Simulink®.

Secondly, according to the established model, the basic physical characteristics are discussed, the relevant conclusions are reached. To the controller design of body angle and the wheel speed, the optimized double PID closed-loop controller with better control effect is built, and the robustness of which is verified by simulation experiment.

In the third part, the device of SBTWR is constructed by using LEGO® MINDSTORMS® NXT™, which includes five parts: gyroscope, accelerometer, controller, power supply module and motor with encoder. With the support from Simulink®, the Model-Based Design (MBD) technique is used to develop the control algorithm for SBTWR.

Finally, the control algorithm is converted to program code by code generator and written into the control chip. A series of experiments are taken, and the results prove the controller designed in this report effective for real system too.

Keywords: underactuated, self-balancing two-wheel robot, system modelling, PID control, Model-Based Design, MATLAB®, Simulink®.

Acknowledgement

When the final report is about to complete, my heart is filled with gratitude.

First of all, I would like to thank my supervisor, Dr Xuewu Dai. Dr Xuewu Dai not only raised the idea of this project, but also gave me a lot of guidance and valuable advices in project developing and report writing, so that the project and report can be proceeded smoothly.

Also, I would like to thank all my tutors who taught me a lot about electrical and electronic engineering. They let me have enough relative knowledge to handle this project.

Then I would like to thank my senior, Nan Cao. His kind assistance helped me to overcome the difficulties I faced in this project and even in life.

I would like to thank my parents, their support is important to me. Their support and care, trust and encouragement, supervision and comfort helps me go through all the honour and frustration, make me not lose myself in prosperity, and not lightly give up in trouble.

Last but not the least, I would like to thank all my friends. When I was under high pressure, they were always able to make me feel relaxed. Thank you all my friends.

This report is dedicated to all the tutors, classmates, friends and relatives who care about me.

Contents

Contents of Figures

Contents of Tables

1. Introduction

1.1 Background

As a kind of urban transportations, automobiles are the important part in people's daily life. And it always represents the level of modern civilization of industry.

However, with the rapid development of automobile industry, the quantity of automobile per capita raises steadily. It causes many problems such as traffic jam and environment pollution.

In order to provide more flexibity and reduce carbon emittion, the concept of self-balancing two-wheel electrical vehicle emerged. It is a kind of clean and small urban electrical auxiliary vehicle, shown in Figure 1, which may provide a solution to those problems caused by traditional transport system.

Figure 1 Segway Project P.U.M.A. Prototype (http://www.segway.com/puma/)

In terms of its characteristics, the self-balancing two-wheel electrical vehicle can be considered as a robot as well. The self-balancing two-wheel robot [1], which is abbreviated to SBTWR, is a non-linear, strong coupling, underactuated and unstable robot system, the relevant results on the control algorithm of which could be applied to other similar systems.

Basically, the SBTWR reads the data of feedback from a gyroscope and an accelerometer to keep vertical balance while it is moving around on two motorized wheels. The controller of this robot handles both driving and vertical balance. The drive control is PID (proportional-integral-derivative) Control based on the modern control theory. The balance control is implemented by a two-wheeled inverted pendulum controller.

1.2 Motivation

Compared with traditional transport system, the two-wheel electrical vehicle takes less area and energy. What's more, the two wheels are controlled by two motors which means it is able to spot turn if set the right motor speed.

With advantages above, the SBTWR is suitable in the narrow place, complicated topography. In other words, if it is in a place where can't drive a car or the ground is too bumping to walk, the SBTWR will be the best choice. Therefore, the SBTWR can be used in a wide range of applications scenarios including supermarkets, airports, golf courses, police patrol, space exploration, smart wheelchairs, high-tech toys, control theory test platform, etc., as the Figure 2 shown below.

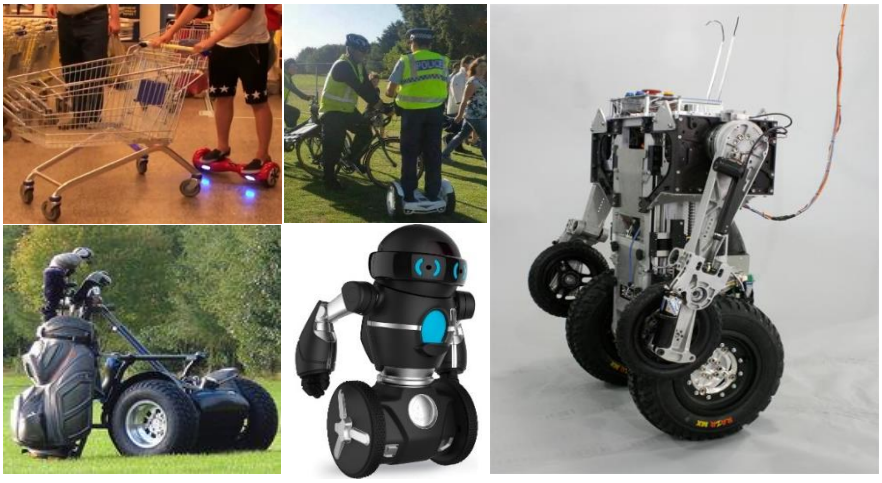

Figure 2 Applications of SBTWR

In addition, the SBTWR is involved in many areas of scientific research, including: mechanisms, micro-computer technology, electronic technology, intelligent control technology, motor drive technology, sensor technology, batteries and materials science.

Thus, the concept of SBTWR system gets attention from different researchers all over the world. Of course, this phenomenon is due to its characteristics of multivariable, strong coupling, time-varying, parameter uncertainty, underactuated, non-linear. As a particular complex system, the SBTWR system is a typical platform used to study different control theory such as uncertain system control, non-linear system control, adaptive control, intelligent control, etc.

As a result, the study of SBTWR is of a strong theoretical and practical significance.

1.3 Project specification

This project is to design a control algorithm which can be applied to a self-balancing two-wheel robot (SBTWR) to make the robot keep self-balancing and be controlled by users or upper computers, as shown in Figure 3.

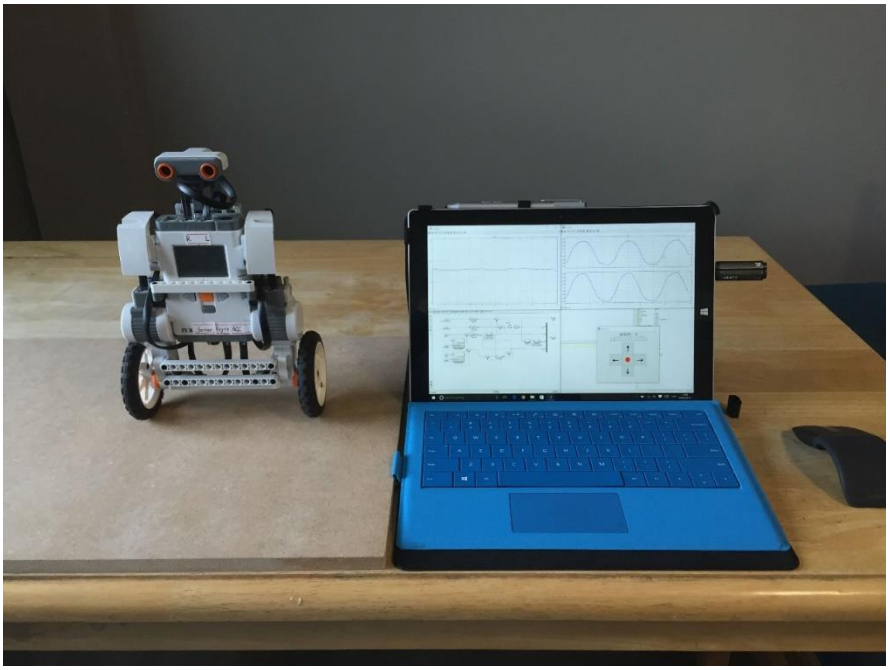

Figure 3 the SBTWR and Control interface in this project

LEGO® supports a well-developed robot platform LEGO® Mindstorms® NXT 2.0. This platform allows users easily construct their robot and use MBD technique to program it by MATLAB® and Simulink® with the relative support package.

In this project, a SBTWR is built by using LEGO® Mindstorms® NXT 2.0 hardware. Then a controller for a SBTWR is designed by using MATLAB® and Simulink®. The main idea is to use feedback from a single-axis gyroscope and a 3-axis accelerometer to maintain vertical balance while the robot moves around on two motorized wheels. Finally, the algorithms is implemented on the SBTWR as standalone, real-time applications.

Once tested the SBTWR at the lab, the same control algorithm can be applied to two-wheel electrical vehicles, such as the Segway.

1.4 Aims and Objectives

The main aim of this project is to design a control algorithm for a SBTWR so that the robot can keep vertical balance while moving around under the user's or upper computers' supervisory instruction.

The major objectives of this project are to:

- Build a SBTWR using with LEGO® MINDSTORMS® NXT 2.0 hardware.
- Build a mathematical dynamic model of this robot.
- Design a SBTWR control system by using MATLAB® and Simulink®.
- Test the SBTWR control system by software simulation in MATLAB®/Simulink®.
- Verify the SBTWR controller by hardware-in-loop (HIP) simulation and experiments at lab.
- Compare and analysis the simulation and experimental Results.
- Development a standalone controller which is applicable to general two-wheel electrical vehicles (such as the Segway).

Depending on the progress and availability of lab device, this project will also try to achieve the following minor objectives:

- Add an ultrasonic sensor to detect the obstacles in front of the SBTWR.
- Add a locator to send the position information of the SBTWR to upper computer.
- Use wireless communication technology to control the SBTWR.
- Design a wireless controller for it.
- Optimize the program to strengthen the robustness of the SBTWR.

2. Literature review and Related technologies

2.1 Basics of two-wheeled inverted pendulum

Inverted pendulum [2] is a complex, unstable, non-linear system, which is an ideal experimental platform used to teach control theory and carry out various control experiments.

The research of inverted pendulum control can effectively reflect many of the typical problems such as non-linear problem, robustness problem, stabilization problem, follow and track problem. A new control method can be tested in inverted pendulum system to verify whether it has a strong ability to deal with non-linear and instability problems. Therefore, the control methods of it have a wide range of uses in the fields of military, aerospace, robotics and general industries, such as the balance control of robotics, rocket flight control and satellite attitude control.

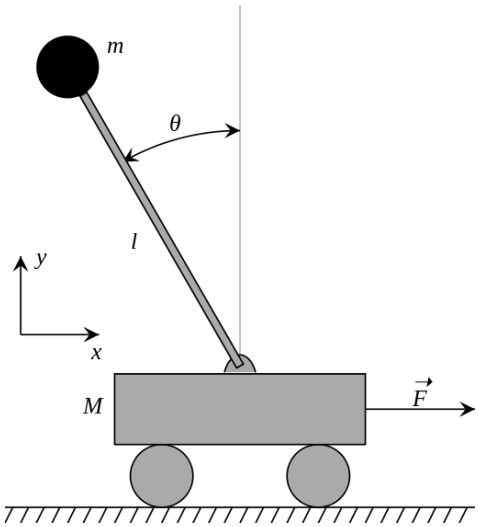

Figure 4 Single inverted pendulum on a cart

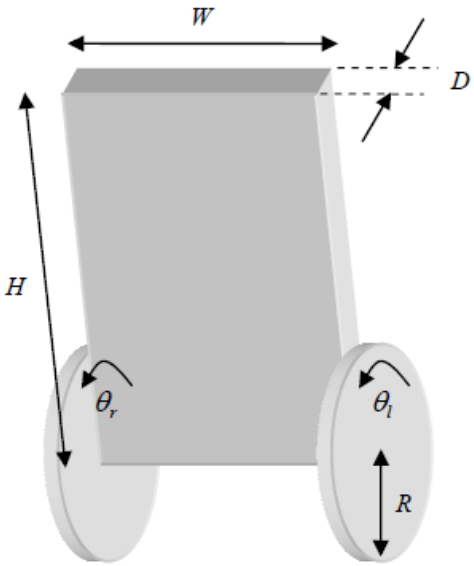

Figure 5 Two-wheeled inverted pendulum

Figure 4 shows a schematic drawing of the inverted pendulum on a cart [ibid]. This is a classic single inverted pendulum model. The rod of it can be considered massless. The mass of the cart and the point mass at the end of the rod are denoted by M and m. The rod has a length l.

With many same characteristics between SBTWR and inverted pendulum, the SBTWR in this report can be equivalent to a single inverted pendulum. That means the control methods of them are similar in many ways.

The main problem to control a SBTWR is to implement its vertical balance as an inverted pendulum. What's more, the speed control and turning control are added to the model. Thus, as shown in Figure 5, the model of a SBTWR is described as a two-wheeled inverted pendulum. Two-wheeled inverted pendulum has two coaxial wheels arranged in parallel, driven by two independent DC servo motors. The inverted pendulum locates above bearings.

2.2 Modelling of the two-wheeled inverted pendulum

It is important to make a mathematical model for a system like SBTWR, which makes the parameters clearly shown in equations for designing and analysing the system in simulation before implementation. Also most control design methods are model based and this means the model will affect the performance of the designed controllers, thus a good model is essential.

In 2002, Felix Grasser, Aldo D'Arrigo, Silvio Colombi and Alfred C. Rufer [3] illustrated how to derive a mathematical model to describe the two-wheel inverted pendulum in their articles. The diagram they used shows in Figure 6.

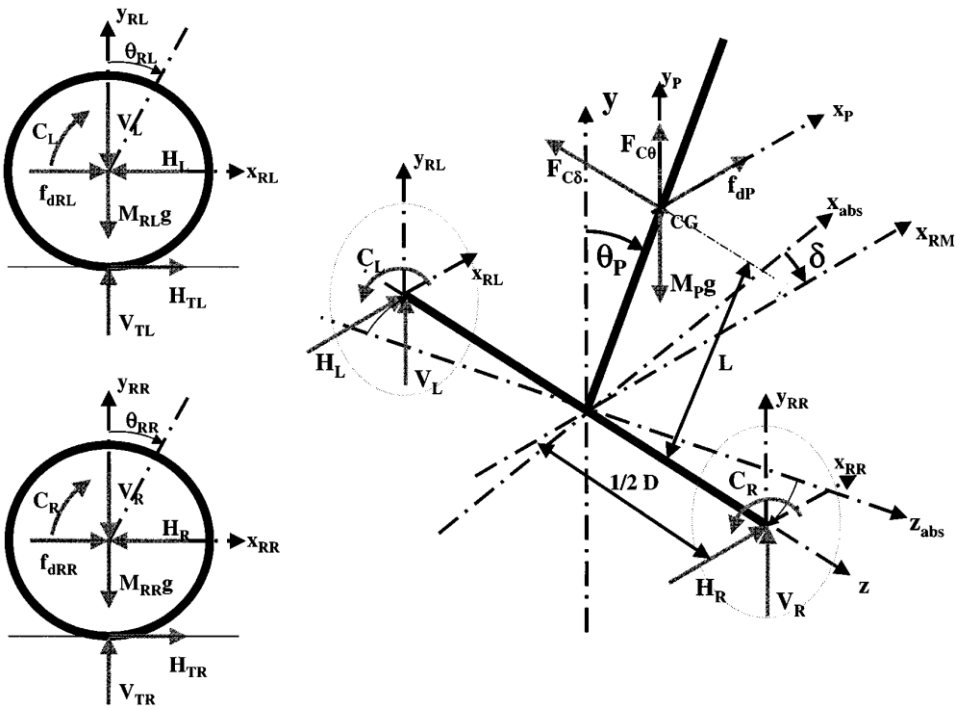

Figure 6 Two-wheel inverted pendulum model

Figure 6 is shown a two-wheel inverted pendulum model based on the parameters characterizing the SBTWR.

A kinematic model cannot adequately describe the motion of a dynamic system. In order to achieve better balance control, speed control and trajectory tracking, the research of dynamic equations of a system is significant. Common mathematical modelling methods for a dynamic system are Lagrangian method and Newton-Euler method.

2.2.1 Motion Equations

The motion equations by the Lagrangian method based on the coordinate system in Figure 7.

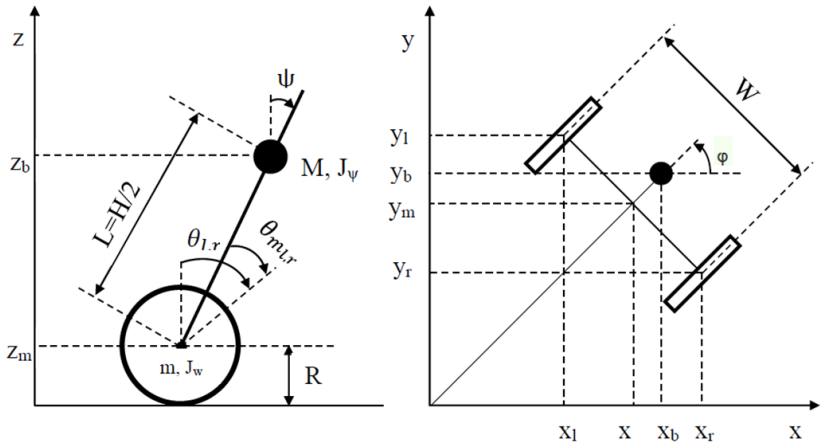

Figure 7 Side view and plane view of two-wheeled inverted pendulum

If the direction of two-wheeled inverted pendulum is x-axis positive direction at t = 0, each coordinates are given as the following.

$$(\theta, \varphi) = \left(\frac{1}{2}(\theta_l + \theta_r), \frac{R}{W}(\theta_r - \theta_l)\right)$$

$$(x_m, y_m, z_m) = \left(\int \dot{x}_m\, dt,\ \int \dot{y}_m\, dt,\ R\right), (\dot{x}_m,\ \dot{y}_m) = \left(R\dot{\theta}\cos\varphi\,, R\dot{\theta}\sin\varphi\right)$$

$$(x_l, y_l, z_l) = \left(x_m - \frac{W}{2}\sin\varphi\,,\ y_m + \frac{W}{2}\cos\varphi\,, z_m\right)$$

$$(x_r, y_r, z_r) = \left(x_m + \frac{W}{2}\sin\varphi\,,\ y_m - \frac{W}{2}\cos\varphi\,, z_m\right)$$

$$(x_b, y_b, z_b) = (x_m + L\sin\psi\cos\varphi\,,\ y_m + L\sin\psi\sin\varphi\,, z_m + L\cos\psi)$$

The translational kinetic energy T_1, the rotational kinetic energy T_2 and the potential energy U are:

$$T_1 = \frac{1}{2}m(\dot{x}_l^2 + \dot{y}_l^2 + \dot{z}_l^2) + \frac{1}{2}m(\dot{x}_r^2 + \dot{y}_r^2 + \dot{z}_r^2) + \frac{1}{2}M(\dot{x}_b^2 + \dot{y}_b^2 + \dot{z}_b^2)$$

$$T_2 = \frac{1}{2}J_W\dot{\theta}_l^2 + \frac{1}{2}J_W\dot{\theta}_r^2 + \frac{1}{2}J_\psi\dot{\psi}^2 + \frac{1}{2}J_\varphi\dot{\varphi}^2 + \frac{1}{2}n^2J_m(\dot{\theta}_l - \dot{\psi})^2 + \frac{1}{2}n^2J_m(\dot{\theta}_r - \dot{\psi})^2$$

$$U = mgz_l + mgz_r + Mgz_b$$

The 5th and 6th terms in T_2 are rotation kinetic energy of an armature in motors. The Lagrangian L is shown below.

$$\mathrm{L} = T_1 + T_2 - U$$

The following variables are used as the generalized coordinates.

θ: Average angle of wheels

ψ: Pitch angle of body

φ: Yaw angle of body

Then the Lagrange equations are the following:

$$\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\partial L}{\partial \dot{\theta}}\right) - \frac{\partial L}{\partial \theta} = F_\theta$$

$$\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\partial L}{\partial \dot{\psi}}\right) - \frac{\partial L}{\partial \psi} = F_\psi$$

$$\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\partial L}{\partial \dot{\varphi}}\right) - \frac{\partial L}{\partial \varphi} = F_\varphi$$

And then following equations can be derived by the 3 equations above.

$$[(2m + M)R^2 + 2J_W + 2n^2J_m]\ddot{\theta} + (MLR\cos\psi - 2n^2J_m)\ddot{\psi} - MLR\dot{\psi}^2\sin\psi = F_\theta$$

$$(MLR\cos\psi - 2n^2J_m)\ddot{\theta} + \left(ML^2 + J_\psi + 2n^2J_m\right)\ddot{\psi} - \mathrm{MgL}\sin\psi - \mathrm{M}L^2\dot{\varphi}^2\sin\psi\cos\psi = F_\psi$$

$$\left[\frac{1}{2}mW^2 + J_\varphi + \frac{W^2}{2R^2}(J_W + n^2J_m) + ML^2\sin^2\psi\right]\ddot{\varphi} + 2\mathrm{M}L^2\dot{\psi}\dot{\varphi}\sin\psi\cos\psi = F_\varphi$$

Moreover, considered torque and viscous friction, the generalized forces of DC motor are given as the following:

$$(F_\theta, F_\psi, F_\varphi) = \left(F_l + F_r, F_\psi, \frac{W}{2R}(F_r - F_l)\right)$$

$$F_l = \mathrm{n}K_t i_l + f_m(\dot{\psi} - \dot{\theta}_l) - f_w\dot{\theta}_l$$

$$F_r = \mathrm{n}K_t i_r + f_m(\dot{\psi} - \dot{\theta}_r) - f_w\dot{\theta}_r$$

$$F_\psi = -\mathrm{n}K_t i_l - \mathrm{n}K_t i_r - f_m(\dot{\psi} - \dot{\theta}_l) - f_m(\dot{\psi} - \dot{\theta}_r)$$

where i_{lr} is the current of DC motor.

The DC motor current cannot be used directly because it is based on PWM control. Therefore, the relation between current i_{lr} and voltage v_{lr} can be evaluated by using DC motor equation. Assuming the friction inside the motor is negligible, the DC motor equations is generally as follows:

$$L_m i_{l,r} = v_{l,r} + K_b(\dot{\psi} - \dot{\theta}_{l,r}) - R_m i_{l,r}$$

What's more, consider that the motor inductance is negligible and is approximated as zero. Therefore the current is:

$$i_{l,r} = \frac{v_{l,r} + K_b(\dot{\psi} - \dot{\theta}_{l,r})}{R_m}$$

From the equation above, the generalized force can be expressed using the motor voltage.

$$F_\theta = \alpha(v_l + v_r) - 2(\beta + f_w)\dot{\theta} + 2\beta\dot{\psi}$$

$$F_\psi = -\alpha(v_l + v_r) + 2\beta\dot{\theta} - 2\beta\dot{\psi}$$

$$F_\varphi = \frac{W}{2R}\alpha(v_l - v_r) - \frac{W^2}{2R^2}(\beta + f_w)\dot{\varphi}$$

$$\alpha = \frac{nK_t}{R_m}, \beta = \frac{nK_tK_b}{R_m} + f_m$$

2.2.2 State Equations

State equations can be derived based on modern control theory. Firstly, consider the limit $\psi \to 0$ ($\sin\psi \to \psi, \cos\psi \to 1$) and neglect the second order term like $\dot{\psi}^2$. The motion equations are approximated as the following:

$$[(2m+M)R^2 + 2J_W + 2n^2J_m]\ddot{\theta} + (MLR\cos\psi - 2n^2J_m)\ddot{\psi} = F_\theta$$

$$(MLR - 2n^2J_m)\ddot{\theta} + \left(ML^2 + J_\psi + 2n^2J_m\right)\ddot{\psi} - \mathrm{MgL}\psi = F_\psi$$

$$\left[\frac{1}{2}mW^2 + J_\varphi + \frac{W^2}{2R^2}(J_W + n^2J_m)\right]\ddot{\varphi} = F_\varphi$$

The first two equations have θ and ψ, the third equation has φ only. These equations can be expressed in the form:

$$\mathrm{E}\begin{bmatrix}\ddot{\theta}\\ \ddot{\psi}\end{bmatrix} + F\begin{bmatrix}\dot{\theta}\\ \dot{\psi}\end{bmatrix} + G\begin{bmatrix}\theta\\ \psi\end{bmatrix} = H\begin{bmatrix}v_l\\ v_r\end{bmatrix}$$

$$\mathrm{E} = \begin{bmatrix}(2m+M)R^2 + 2J_W + 2n^2J_m & MLR\cos\psi - 2n^2J_m\\ MLR - 2n^2J_m & ML^2 + J_\psi + 2n^2J_m\end{bmatrix}$$

$$\mathrm{F} = 2\begin{bmatrix}\beta + f_W & -\beta\\ -\beta & \beta\end{bmatrix}$$

$$\mathrm{G} = \begin{bmatrix}0 & 0\\ 0 & -MgL\end{bmatrix}$$

$$\mathrm{H} = \begin{bmatrix}\alpha & \alpha\\ -\alpha & -\alpha\end{bmatrix}$$

$$\mathrm{I}\ddot{\varphi} + J\dot{\varphi} = K(v_r - v_l)$$

$$\mathrm{I} = \frac{1}{2}mW^2 + J_\varphi + \frac{W^2}{2R^2}(J_W + n^2J_m)$$

$$\mathrm{J} = \frac{W^2}{2R^2}(\beta + f_W)$$

$$\mathrm{K} = \frac{W}{2R}\alpha$$

Then consider the following variables x_1, x_2 as state, and u as input.

$$x_1 = [\theta, \psi, \dot{\theta}, \dot{\psi}]^T, x_2 = [\varphi, \dot{\varphi}]^T, \mathrm{u} = [v_l, v_r]^T$$

Consequently, the state equations of two-wheeled inverted pendulum can be derived from:

$$\dot{x}_1 = A_1 x_1 + B_1 u$$

$$\dot{x}_2 = A_2 x_2 + B_2 u$$

$$A_1 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & A_1(3,2) & A_1(3,3) & A_1(3,4) \\ 0 & A_1(4,2) & A_1(4,3) & A_1(4,4) \end{bmatrix}, B_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ B_1(3,1) & B_1(3,2) \\ B_1(4,1) & B_1(4,2) \end{bmatrix}$$

$$A_2 = \begin{bmatrix} 0 & 1 \\ 0 & -J/_I \end{bmatrix}, B_2 = \begin{bmatrix} 0 & 0 \\ -K/_I & K/_I \end{bmatrix}$$

$$A_1(3,2) = -gMLE(1,2)/det(E)$$

$$A_1(4,2) = gMLE(1,1)/det(E)$$

$$A_1(3,3) = -2[(\beta + f_w)E(2,2) + \beta E(1,2)]/det(E)$$

$$A_1(4,3) = 2[(\beta + f_w)E(2,2) + \beta E(1,2)]/det(E)$$

$$A_1(3,4) = 2\beta[E(2,2) + E(1,2)]/det(E)$$

$$A_1(4,4) = -2\beta[E(1,1) + E(1,2)]/det(E)$$

$$B_1(3,1) = B_1(3,2) = \alpha[E(2,2) + E(1,2)]/det(E)$$

$$B_1(4,1) = B_1(4,2) = -\alpha[E(1,1) + E(1,2)]/det(E)$$

$$\det(E) = E(1,1)E(2,2) - E(1,2)^2$$

2.3 PID Controller

PID controller (proportional - integral - derivative controller) [4] is a widely used controller composed by the proportion unit P, integral unit I and differential unit D. This controller can be set by changing the parameters K_P, K_I and K_D. The block diagram of a PID controller shows in Figure 8.

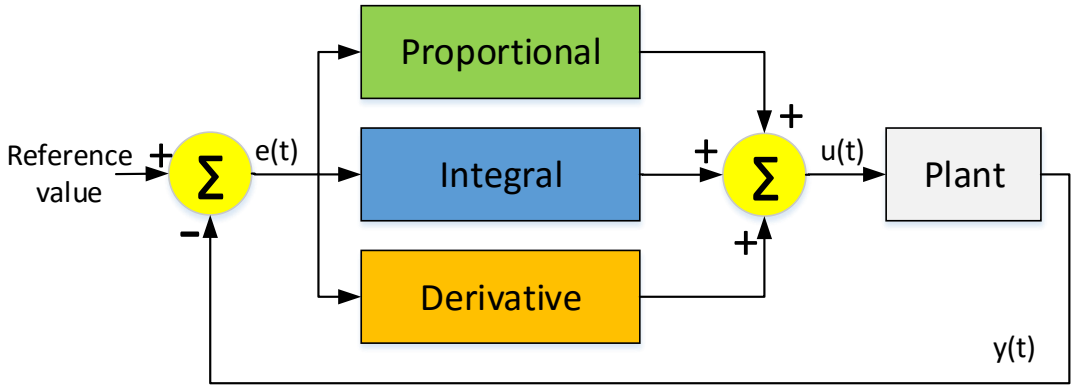

Figure 8 Block diagram of PID controller

Here is the basic equations of PID controller:

$$u(t) = K_p e(t) + K_I \int_0^t e(\tau)\, d\tau + K_D \frac{de(t)}{dt}$$

Where

$e(t) = Reference\ value - y(t)$

$t --current\ time$

$\tau --integral\ variable, from\ 0\ to\ the\ current\ time\ t$

$K_P --proportional\ gain, adjustable\ parameter$

$K_I --integral\ gain, adjustable\ parameter$

$K_D --derivative\ gain, adjustable\ parameter$

Basically, PID controller subtracts the gained data from the reference value, then the difference is used to calculate the new input values, the purpose of this new input value is to let the data system to reach or remain the reference value. The input values can be adjusted according to the rate of emergence of historical data and differences, making the system more accurate and stable.

PID controller is mainly applied to the system which is substantially linear, and the dynamic characteristics of the system are stable. What's more, it is a common feedback loop part in industrial control applications.

2.4 Gyroscope and Accelerometer

The gyroscope and accelerometer used in this project are MEMS (Micro-Electro-Mechanical Systems) sensors, shown in Figure 9.

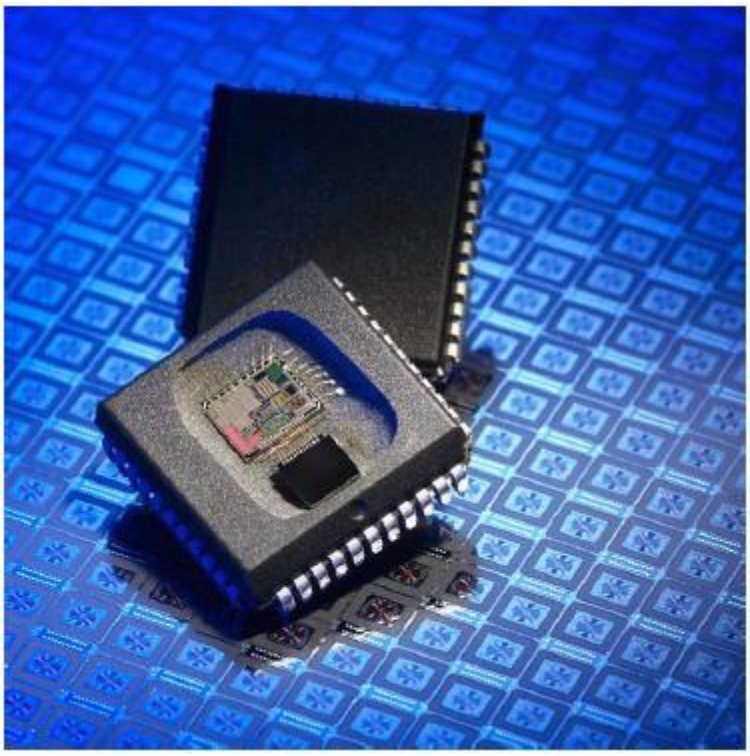

Figure 9 MEMS sensors

MEMS can not only collect, process and transmit information or instructions, but also able to act in accordance with an external command in accordance with the information acquired or autonomously. It uses microelectronics and micro-processing technology (including bulk silicon micro-machine, silicon surface micro-machine, LIGA and wafer bonding techniques) combined manufacturing process, manufacturing a variety of excellent performance, low cost, miniaturized sensors.

The MEMS Gyroscope usually contain two directions' movable capacitor plates. Radial capacitor plate is applied oscillation voltage to force the object moving radially, while horizontal capacitor plate is used to measure due change in capacitance caused by the lateral movement. Because the value measured is proportional to the angular velocity, so the change in capacitance can be calculated the angular velocity.

Similarly, the MEMS Accelerometer in this project uses a capacitor as an interface to detect the micro-displacement caused by inertia force from the inertial mass. The mass is attached to the substrate via an elastic micro-beam, one capacitance detection plate is generally arranged on the mass movement, and the other plate is disposed on the fixed substrate. With high sensitivity and accuracy, good stability, small temperature drift, low power consumption, and strong overload capability, the capacitive micro-accelerometer can use static electricity to achieve closed-loop feedback control, significantly improving the performance of the sensor.

2.5 Model-Based Design

Yorihisa Yamamoto [5] shows the benefits of model-based design in 'NXTway-GS Model-based Design'.

Generally, Model-Based Design (MBD), is a development technique of software that utilize simulation models to design the program. In control systems, a designer models a plant and a controller or part of them, and tests the control algorithm based on a computer simulation or real-time simulation. The real-time simulation makes it possible to test the algorithm in real time, with the help of code generator to generate C code from the models. While in Rapid Prototyping (RP), a controller is replaced by a real-time simulator, and Hardware in the Loop Simulation (HILS) is the Rapid Prototyping in a plant version.

Furthermore, auto code generation products [6] enables us to generate C program for embedded controllers (microprocessor, DSP processer, etc.) from the controller model. Figure 10 [5] shows the MBD concept for control systems based on MATLAB® products and Simulink® products.

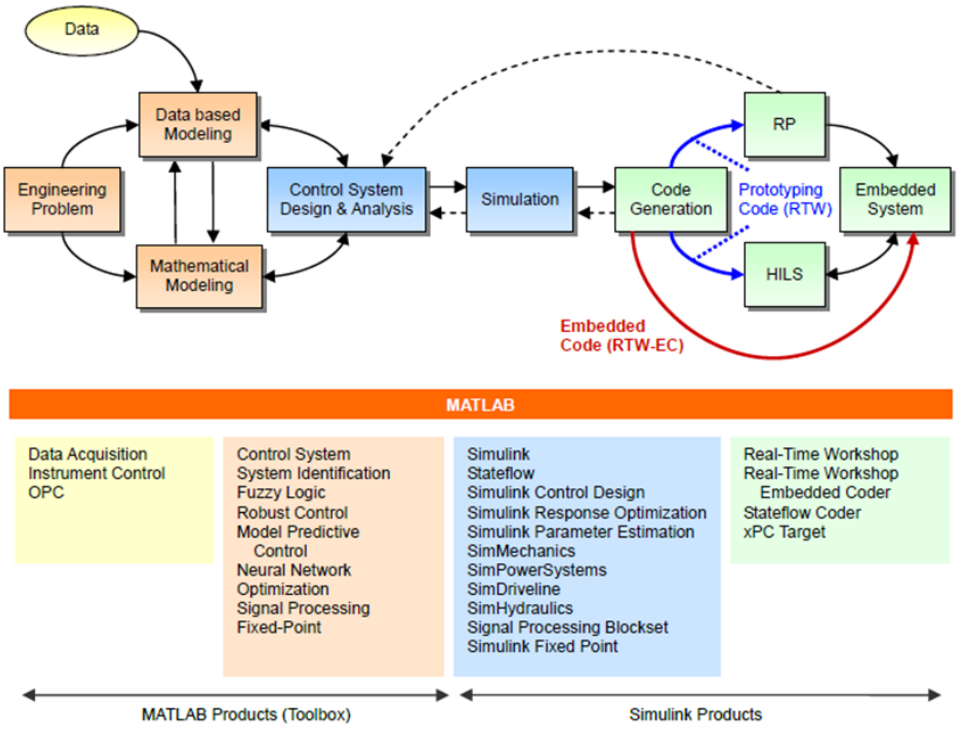

Figure 10 MBD for control systems based on MATLAB and Simulink [5]

MBD allows researchers to design more useful and functional control systems intuitively rather than getting confused by hundreds lines of C code. Thus, it is a better way of controller designing compared with those traditional ways.

3. System Design

In this report, the LEGO® MINDSTORMS® NXT™ is used to develop a hardware platform and algorithms for SSBTWR research and post-commissioning and other issues. It is proposed to build a model based on the following algorithm in MATLAB/Simulink environment and automatically generate C code. Finally, through the serial port the programs is downloaded to the robot and executed by the controller.

According to the idea above, the whole system is divided into two parts: the host (PC) and the SBTWR object. Wherein the SBTWR object contains the following elements: controller, sensors, motors.

The host based on PC runs graphical programming software and can process controlled object modelling and design control algorithm. It will eventually generate an executable file, and through the serial port communication the file will be downloaded to the target controller.

In the experimental system, the first to do is to model the object. In MATLAB/Simulink modelling environment, it is easy to use existing MATLAB math toolboxes, supporting the completion of the control algorithm. After modelling is completed, the model will be compiled to generate executable code, which will be downloaded to the target controller. Then the controller will control the motors as expected.

The system structure is shown below.

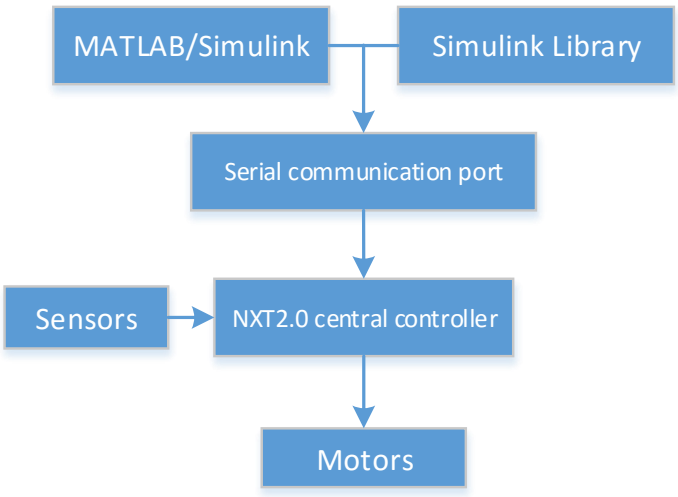

Figure 11 System structure diagram

4. Hardware Design

4.1 Robot construction

As shown below, the SBTWR in this project is built with NXT Robot set and HiTechnic accelerometer and gyroscope. It can be used for adjusting the program and debugging. The basic structure of this robot is referred to NXTway [7]. To get a better reliability and stability, the overall structure is optimized by reducing the useless parts and adding protection parts. What's more, the ultrasonic sensor is added as an additional feature.

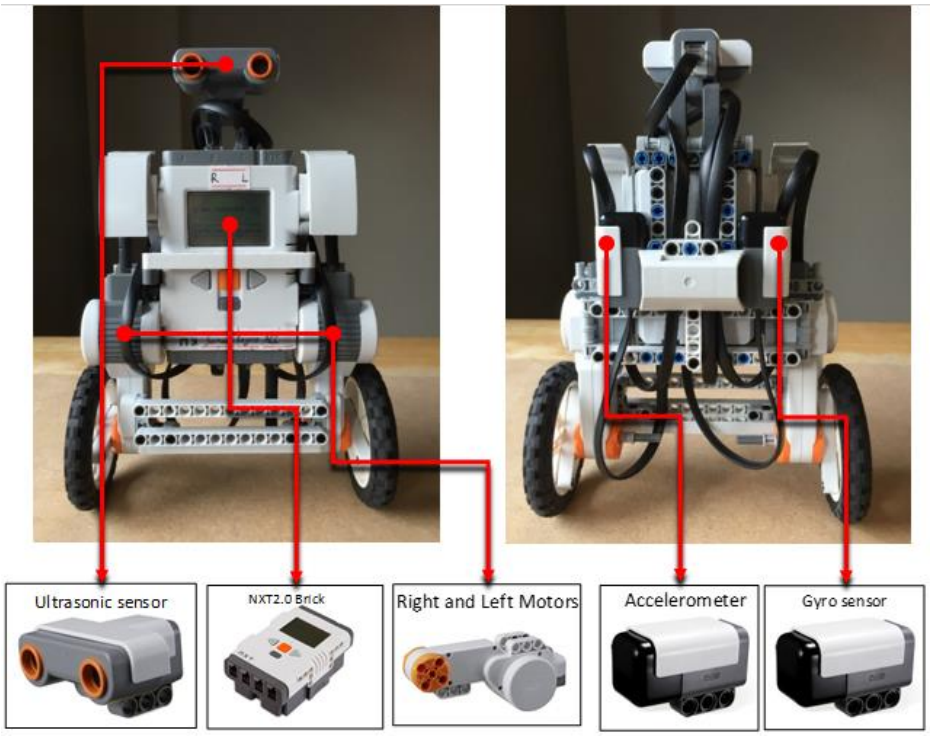

Figure 12 Front view and back view of SBTWR

In addition, tilt sensors, including a gyroscope and accelerometer, should be set up near the centre of mass of the robot to decrease the effect to measuring accuracy caused by the tremble when the robot is moving.

In order to avoid the dumping breaking the shell and the circuit board when the SBTWR is moving. When the mechanical structure is designed, it need to be considered to equip before and after the SBTWR with a bracket or a crash cushion. Once dumping or uncontrolled, they can protect the SBTWR.

4.2 Schematic Diagram

The connection diagram is shown below. The port A, B, C are the output and input ports of NXT2.0 Brick [8], while the port 1 to 4 are only the input ports. So the motors are connected to PORTB and PORTC for controlling the motors and getting data from the internal encoder in motors. The accelerometer, gyroscope and ultrasonic sensor are connected to PORT4, PORT5, PORT6 because they can only supply data to NTX2.0 Brick. Moreover, the NXT2.0 Brick, motors, sensors of LEGO are all modularized, it is convenient to construct a robot with them.

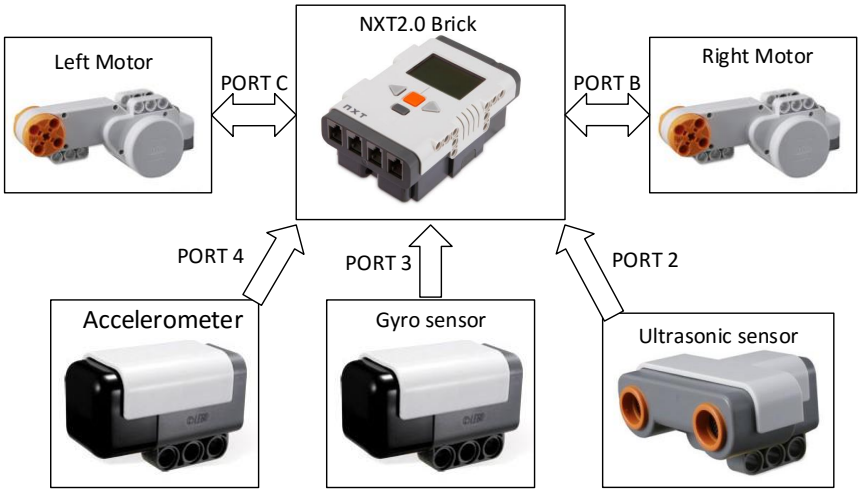

Figure 13 Connection diagram

The overall schematic diagram is shown in Figure 14. The SBTWR gets the feedback data from a gyroscope and an accelerometer to keep the upright balance of the robot. The controller of this robot handles both speed and body angle of the robot.

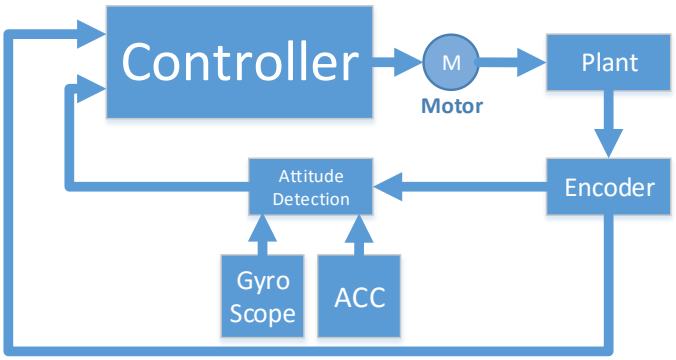

Figure 14 Schematic diagram of SBTWR

4.3 Motors and Sensors

Figure 15 shows the servo motor of LEGO NXT, the power and running speed of it can be controlled by set the value of input. With an internal encoder, it can be easily used. However, a disadvantage of this motor is the gears are loose because they are all plastic. This problem can be correct by adjust the program. Moreover, in real use, the other problem appeared, which is the speeds of two motors are different though the input values are same. By adding the calibration algorithm, this problem is fixed well.

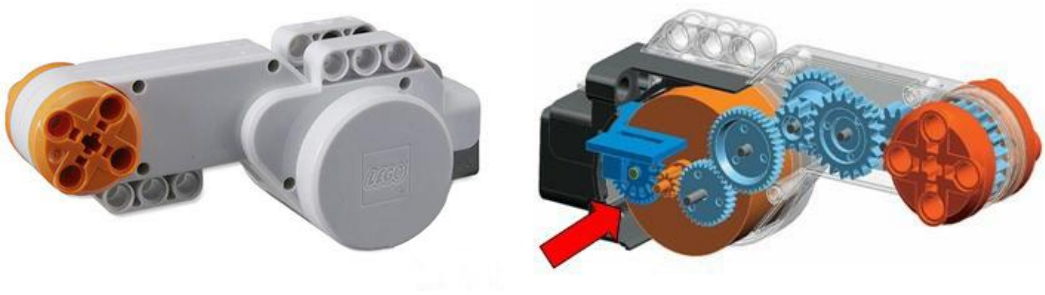

Figure 15 NXT servo motor

The HiTechnic accelerometer and gyroscope is shown in Figure 16. They are from a third party company called HiTechnic. The appearances of them look same because they are packaged in same shaped plastic shell.

Figure 16 HiTechnic gyroscope and accelerometer

4.4 NXT Central Controller

NXT2.0 Brick [8], shown in Figure 17, is the core part of the SBTWR in this project. It consists LCD, buttons and battery.

Figure 17 NXT2.0 Brick

The main processor of it is Atmel ARM7, with the assistant of the co-processor Atmega48. As the performance of NXT2.0 Brick is beyond the requirement of the central controller of SBTWR, more functions and features can be developed in NXT2.0. The schematic diagram of NXT2.0 Brick is shown below.

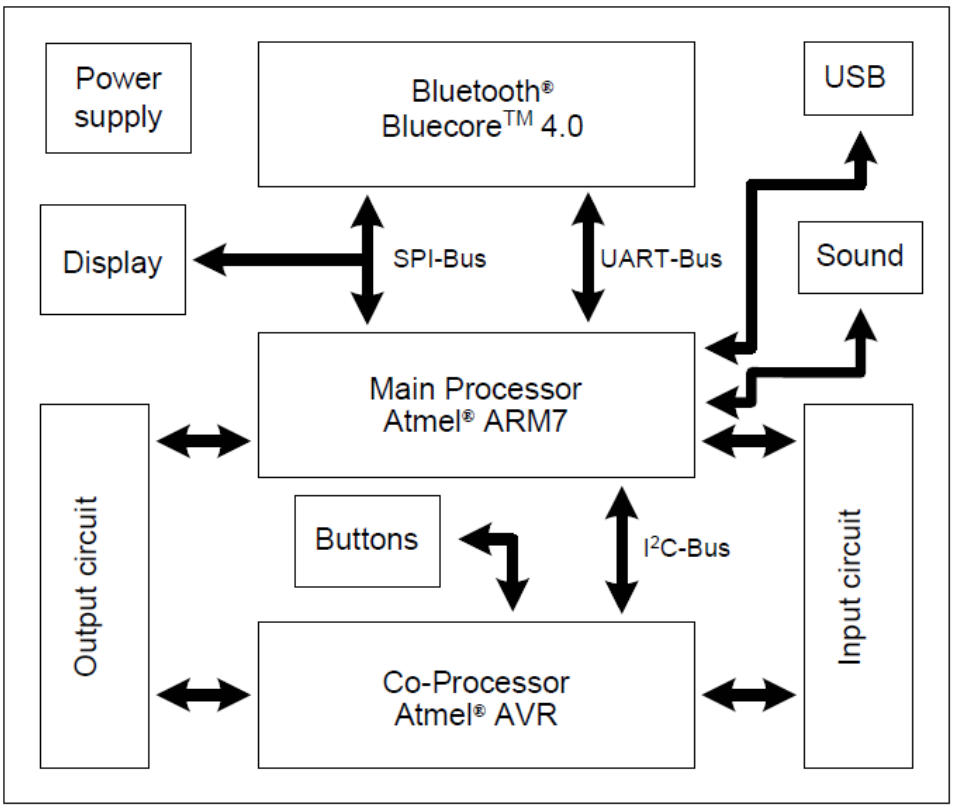

Figure 18 Hardware block diagram of the NXT brick

4.5 Physical parameters

The physical parameters shown below will be used to design a simulated robot in Matlab by using mathematic model to calculate the basic value of control parameters and do control simulations.

Name	Value	Measurement unit	Description
g	9.81	m/sec^2	Gravity acceleration
m	0.03	kg	Wheel weight
R	0.04	m	Wheel radius
J_w	$mR^2/2$	Kgm^2	Wheel inertia moment
M	0.6	kg	Body weight
W	0.14	m	Body width
D	0.04	m	Body depth
H	0.144	m	Body height
L	H/2	m	Distance of the centre of mass from the wheel axle
J_ψ	$ML^2/3$	kgm^2	Body pitch inertia moment
J_φ	$M(W^2+D^2)/12$	kgm^2	Body yaw inertia moment
J_m	1×10^{-5}	kgm^2	DC motor inertia moment
R_m	6.69	Ω	DC motor resistance
K_b	0.468	V sec/rad	DC motor back EMF constant
K_t	0.317	Nm/A	DC motor torque constant
n	1		Gear ratio
f_m	0.0022		Friction coefficient between body and DC motor
f_w	0		Friction coefficient between wheel and floor

Table 1 Physical parameters of the SBTWR

- The value of J_m, n, f_m, f_w are difficult to measure so they are the approximate values.

5. Software Design

5.1 Control system design

5.1.1 Software features and frame

It is essential to divide the functions into several features, which can make the idea of the whole algorithm simple and clear.

The main features of the software include:

(1) Each sensor's signal acquisition, processing.

(2) Motor PWM output.

(3) Robot Operation Control: upright control, speed control, direction control.

(4) Robot running processes control: initialization, start and end, robot status monitoring.

(5) Robot information display settings and parameters: status display, PC monitor and parameter settings.

Since the main features of the program have been determined, the idea of the algorithm is shown above.

The flowchart of program is shown below.

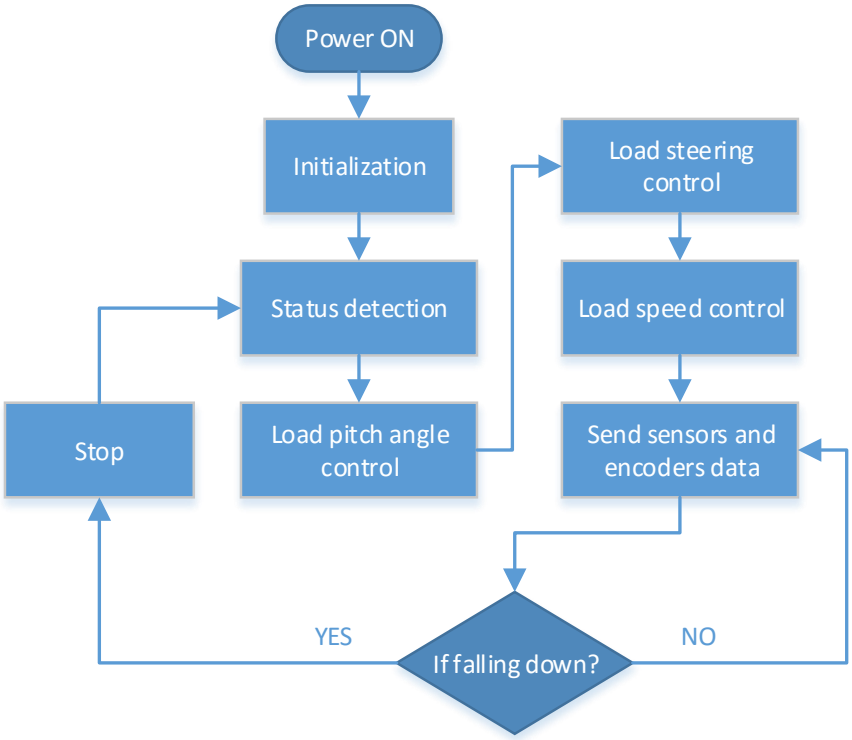

Figure 19 Flowchart of main program

From the flowchart of main program in Figure 19, it is clear to get the sequence of the program.

After power on, the initialization will be carried out. Initialization process includes two parts, one for each application's initialization. The other part is the program initialization, where the variables for the control program applied will be initialized.

After initialization, the first is to detect the robot status for 4 seconds. By reading the accelerometer values the program determine the values when the robot in an upright position. After that, the pitch angle control, steering control and speed control will be loaded to keep body balance of the robot.

In the main program loop continuously send monitoring data sent to the host computer through the serial port to monitor. Besides, it checks whether the robot fall. Analysing fall can be determined by reading the attitude data from sensors. If the robot fell down, the running of it will be stopped. Then re-enter the robot status detection process.

Moreover, those closed loop control programs are all managed by global variables.

5.1.2 Main algorithm

Figure 20 is shown the basic control system diagram.

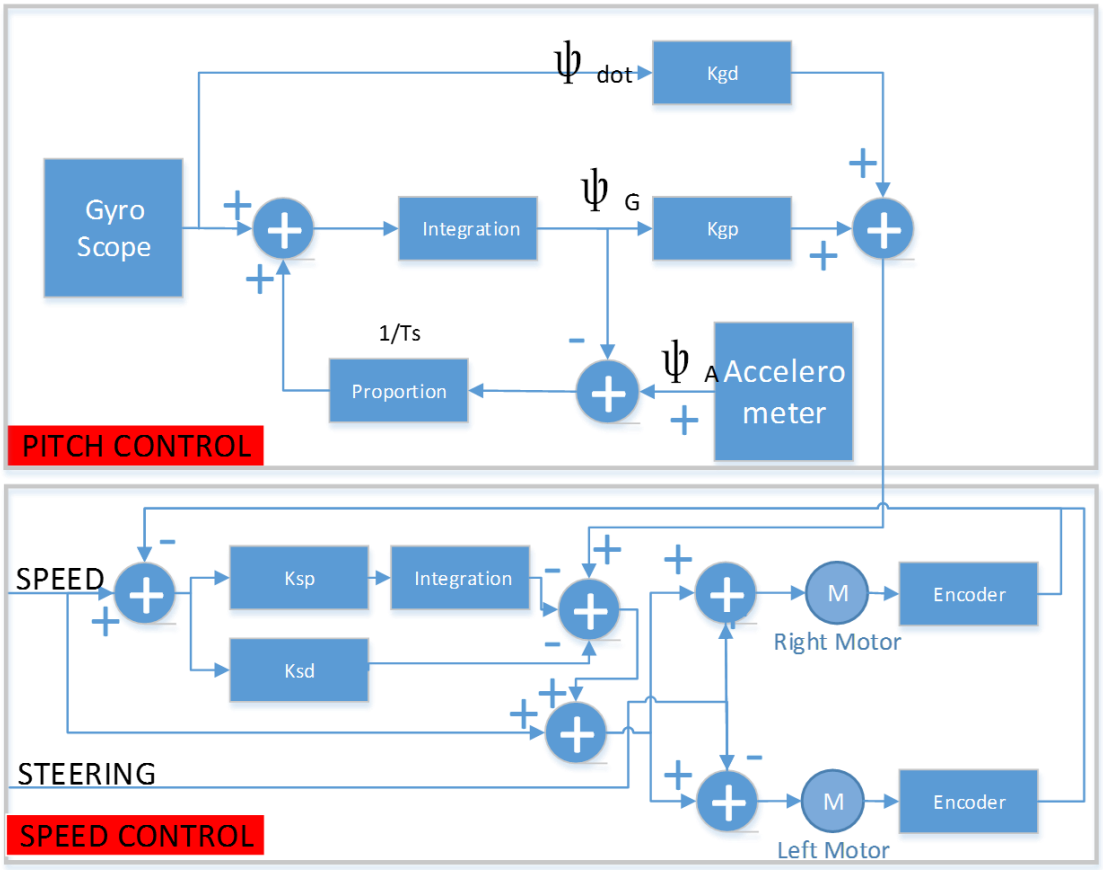

Figure 20 Basic Control System Diagram

The control functions include:

(1) Angle Calculate: angle calculation function. The robot calculates the angle and angular velocity according to the value collected from gyroscopes and gravity sensors. This function is called once every 4 milliseconds.

(2) Angel Control: Robot upright control functions is used to calculate control amount of the motor depending on the angle and angular velocity. It is called once every 4 milliseconds.

(3) Speed Control: Robot speed control function. According to the motor speed and speed reference, the control amount of the motor is calculated. This function is called once every 4 milliseconds.

(4) Speed Control Output: Output speed smoothing function.

(5) Direction Control: directional control functions. According to the value of about two car models to capture electromagnetic sensors to calculate the amount of the angle control. This function is called once every 4 milliseconds.

(6) Direction Control Output: Output direction control smoothing function.

(7) Motor Output: Motor output collection functions. According to angle control, speed control and direction control, the control amount is obtained by superposition, respectively getting the amount of two output voltage. The superimposed output will be put into the saturation process. This function is called every 4 milliseconds. Please note that at this amount superimposed by negative polarity.

(8) Motor Speed Out: Motor PWM output calculation function. According to the polarity of the output amount of the left and right motors, superimposed on a small dead zone value, this function is used to overcome the mechanical static friction of robot. It is called every 4 milliseconds.

(9) Set Motor Voltage: PWM output function: according to the output of the two motors, calculate the value of the PWM control register.

The nine functions above are all called every 4 milliseconds, so the sample time of each module should be set to 0.004s in Simulink, shown in Figure 21.

Figure 21 Sample time of the blocks

The parameters are passed between functions through calling functions for parameters variables or global variables. All parameters are single precision floating point (float). Angle Calculate and Speed Control integral calculation, these two functions are also integral to retain two global variables. The speed pulse information and analogue information collected through the accumulation are averaged to provide state variables of the robot for the angle, speed and direction control.

5.1.3 Implementation

This section will explain the main control functions. The algorithm can be roughly divided into two sub-sections, Angle Control and Speed Control.

When controlling the speed, it is required to remain the robot upright. Similarly, when controlling the angle, it also need to keep the speed and steering steady. These two tasks are to maintain the balance of the robot. Since the robot is effected by two control functions. The changes of speed and angle should be kept smooth to reduce the balance control interference. With speed adjustment, for example, by changing the balance of the robot tilt control setpoint, thereby changing the inclination angle. In order to avoid the effects of balance control, the robot is required to change inclination very slowly.

The implementation of angle control and speed control is shown below.

(1) Angle Control

This program reads the data output value from gyroscope, and then calculate the inclination of the robot.

The values from gyroscope and accelerometer need to subtract the zero point drift value and angle initial value. This value can be determined in the initialization. When the robot remains upright stable, the values of gyroscope and accelerometer are the zero point drift value and angle initial value. These values will be with some errors, which can be eliminated by the speed control function.

The data from accelerometer can be normalized to between -90 to +90 by using the Trigonometric and hyperbolic functions of Simulink. Similarly, the data from gyroscope need to be normalized by using a factor. The robot angle and angular velocity multiplied by respective coefficients can be upright control output to speed control,

The block diagram of this function is shown below.

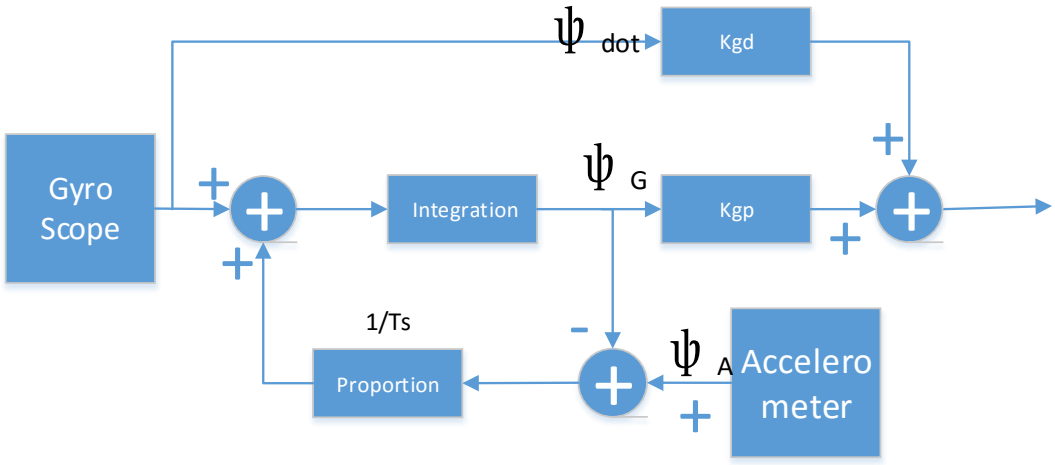

Figure 22 Angle calculation Function

After normalizing, the function can be implemented by the block diagram above, where

ψ_{dot} -- angular speed value from Gyroscope

ψ_G -- angle value from integration of ψ_{dot}

ψ_A -- angle value from accelerometer.

The parameters kgd and kgp can be calculated by mathematic model or experiments. The angle control function uses PD control. Actually the accelerometer is not necessary to be added because the speed control can also decrease the gyro drift. However, to enhance the robustness of the SBTWR in this project. Both the accelerometer and speed calibration will be used.

(2) Speed Control

By reading the data from encoder the speed of the robot can maintain in speed reference by using the PD control algorithm shown below. The parameter ksp and ksd can be determined by mathematic model or experiments. The angle control output is added to decrease the error caused by data reading of gyroscope and accelerometer.

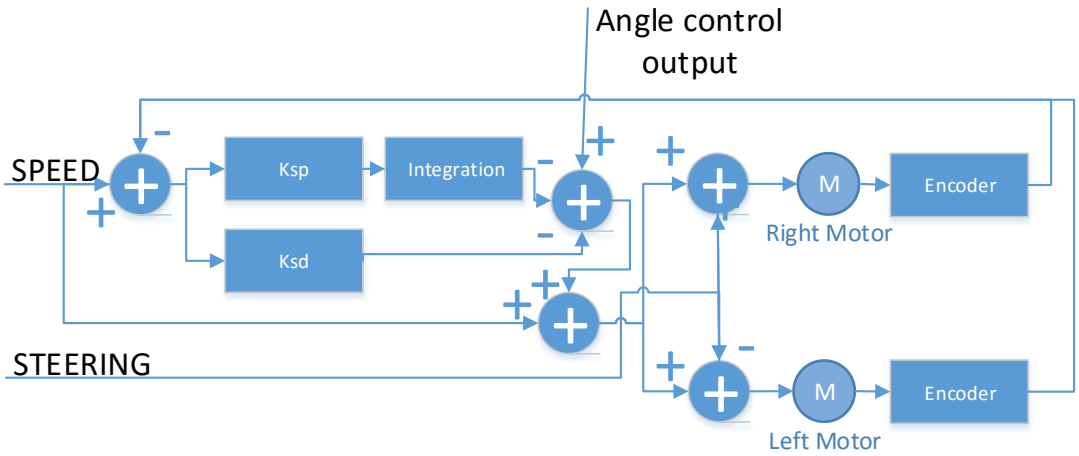

Figure 23 Speed and Steering Control

The steering control is implemented by adding a different value to the motor, which makes the robot can spot turn. The turning speed need to be set to a suitable value. Also, there should be a function to stop the Speed Control when steering in case of the error cause by the collision.

5.2 Simulink Graphic Program

5.2.1 Hardware (Plant)

The support package for LEGO® Mindstorms® NXT is a set of modularized block in Simulink Library, used to develop graphic program. The blocks are shown below.

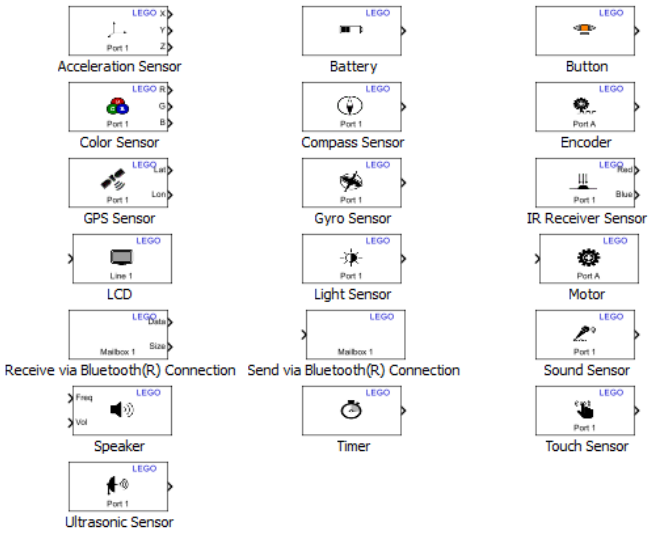

Figure 24 The Simulink Blocks of support package for LEGO Mindstorms NXT

With the support package above, the hardware subsystem can be constructed, shown below.

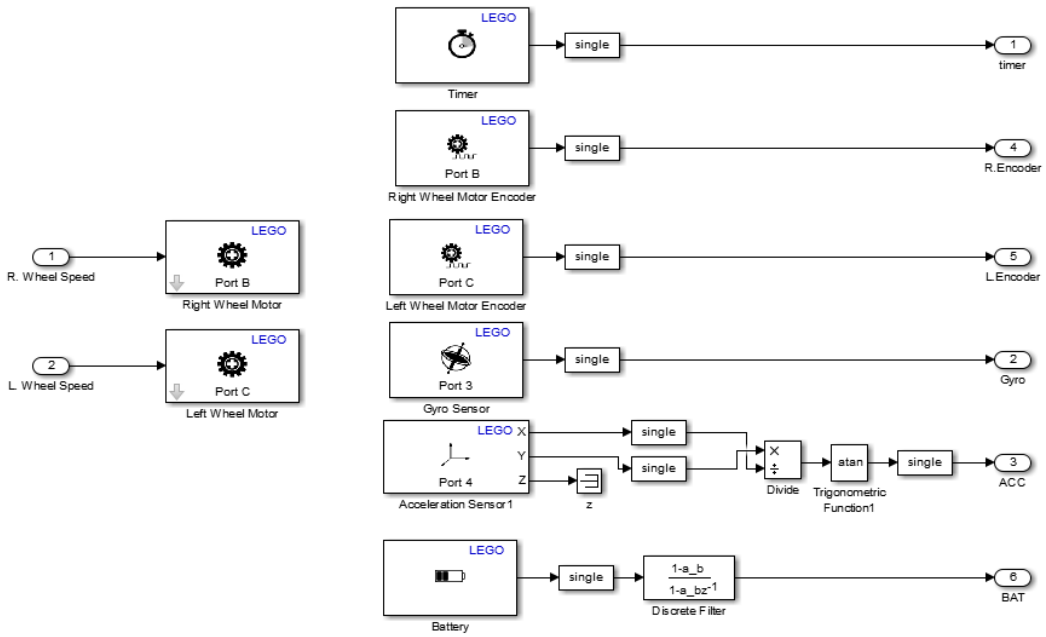

Figure 25 Hardware subsystem in Simulink

In hardware subsystem, the SBTWR is constructed, shown in Figure 25. the input are right and left speed value, while the output are Timer, right and left motor Encoders, Battery output value, Gyroscope and Accelerometer. Moreover, the output of gyroscope is A/D value (0-1023), shown in Figure 26, and the output of accelerometer is the gravity acceleration value of X, Y, Z axis. In this project, only the X and Y axis is used to calculate the body angle, so the ACC output of the hardware subsystem is pre-calculated, shown in Figure 27. According to 5.1.2, the sample time of each blocks should be set to 0.004.

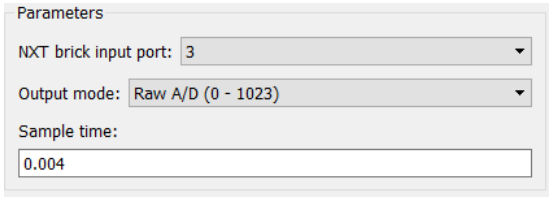

Figure 26 the Parameters of Gyro sensor

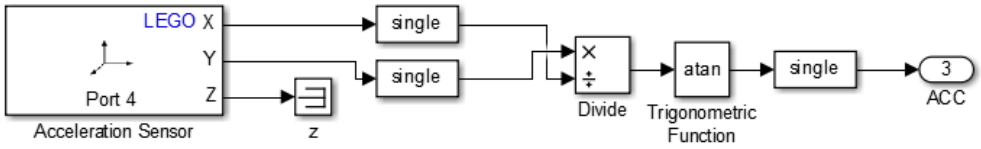

Figure 27 Acceleration output function

According to the description of this block, the Motor Encoder outputs the rotation angle in degrees. This value is integrated so it may be more than 360 degrees. Select the NXT brick output port parameter to which this sensor is connected. Use Reset mode parameter to specify when to reset the rotation measurement. When Reset mode parameter is set to reset by external signal, the block takes an int8 value from input port to trigger the reset.

The battery module outputs battery voltage in millivolts (mV), approximately 9000 mV when fully charged. The output value of it includes noise so there is a discrete filter of direct form 2 used to filter the output. The parameter of the filter is shown below.

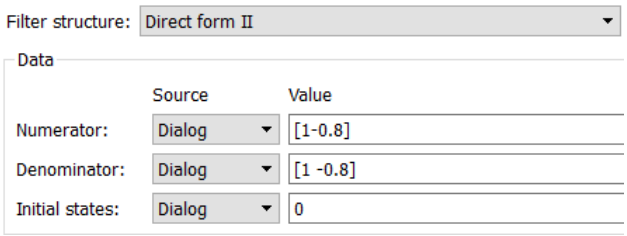

Figure 28 Discrete Filter Parameters

According to the description, the Timer module outputs the time in milliseconds (ms) since enabled. It is used to time the 4 seconds initialization, which used to measure the zero point drift of gyroscope and the initial angle of the robot.

The motor block controls the power and turning direction of the motor. The input port value can range from -100 to +100. When the block input value is zero, it can be selected to brake it or not.

After developing the hardware subsystem, the program can be created as a block, shown below.

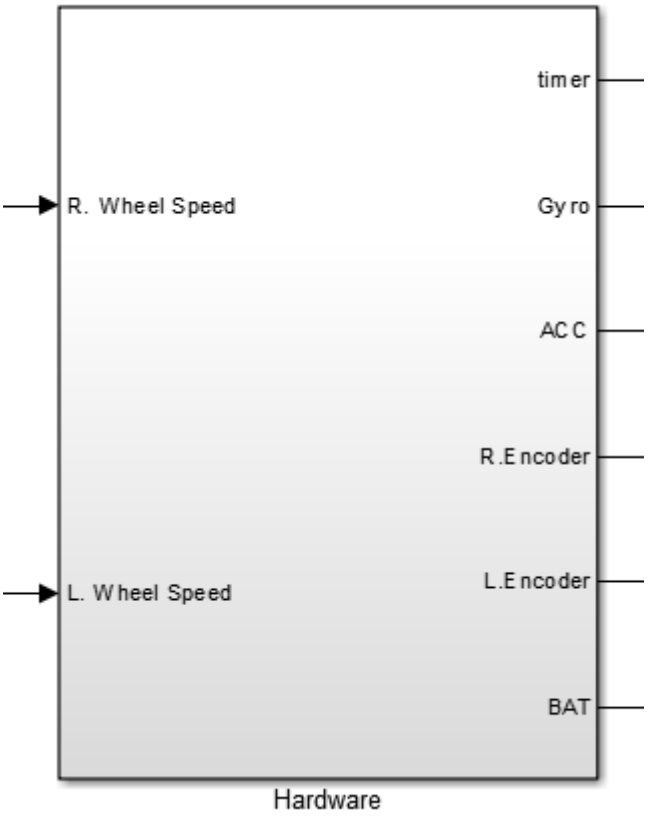

Figure 29 Hardware subsystem block

The hardware subsystem is essential for the whole program, it is not only the part of the program, but also showed the robot structure clearly. In model-based design, a hardware model can ensure developers know about the devices directly.

5.2.2 Initialization

Before control the robot, the initial attitude status of it should be determined. The initialization subsystem is used to detect the zero point drift value of gyroscope and the initial angle of accelerometer, shown below.

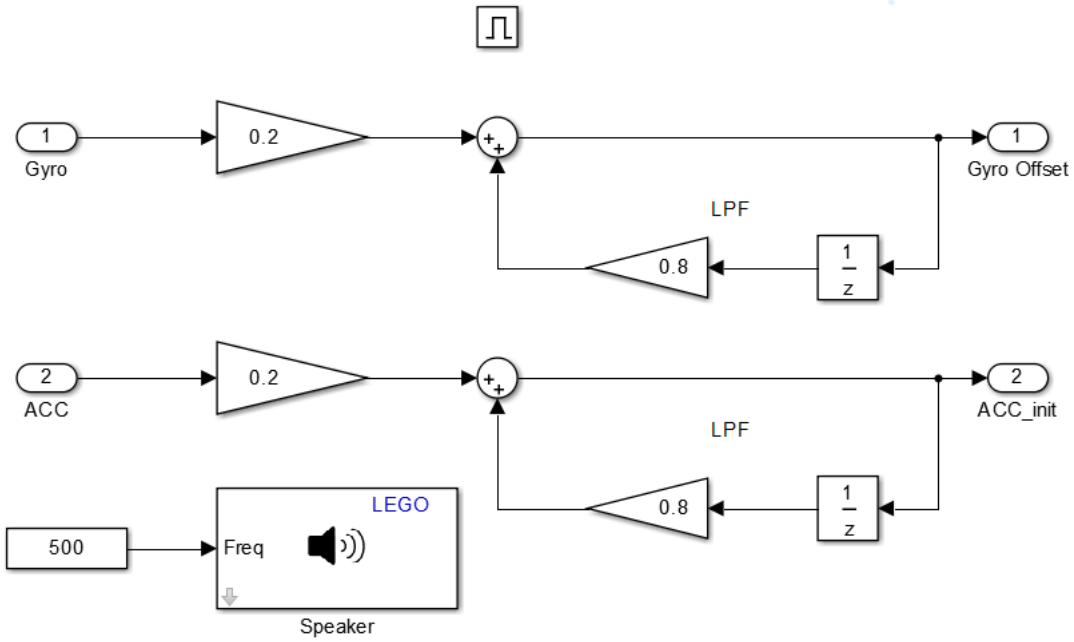

Figure 30 Initialization subsystem

This is an enabled subsystem, in the first 4 seconds after power on, this subsystem will be enabled to get the initial values and beeping. Meanwhile, the robot should be hold by hand to keep an upright and balance state, when the beeping is off, loose the hand and the robot will run the controller and keep vertical balance.

The timer is the one in Hardware subsystem and the function is shown below.

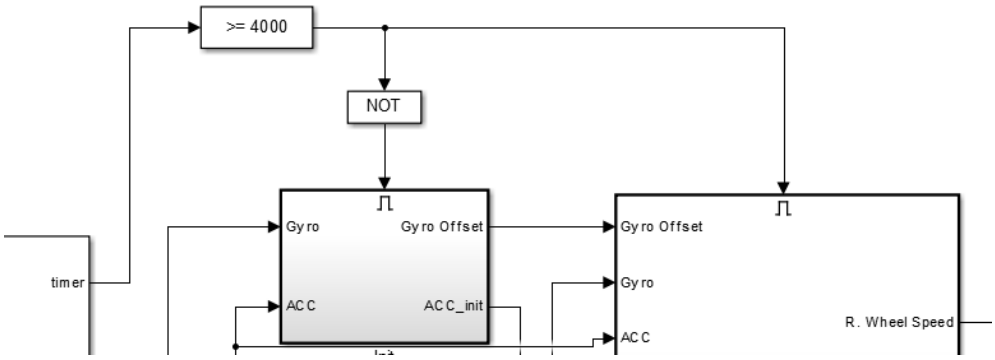

Figure 31 Time measurement function

5.2.3 Controller

The Figure 32 above is shown the controller subsystem. It reads the sensors and encoders input to calculate the speed values sent to motors.

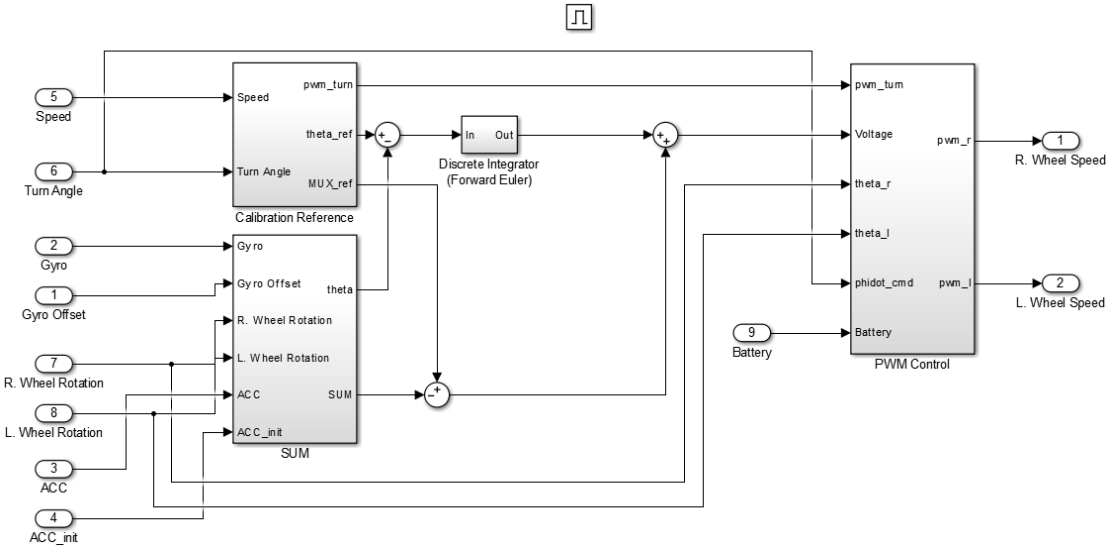

Figure 32 Controller subsystem in Simulink

This subsystem is with enable function, it will load when the enable signal is 1. At that time, the beeping of the speaker is over.

The Calibration Reference subsystem, is shown in Figure 33.

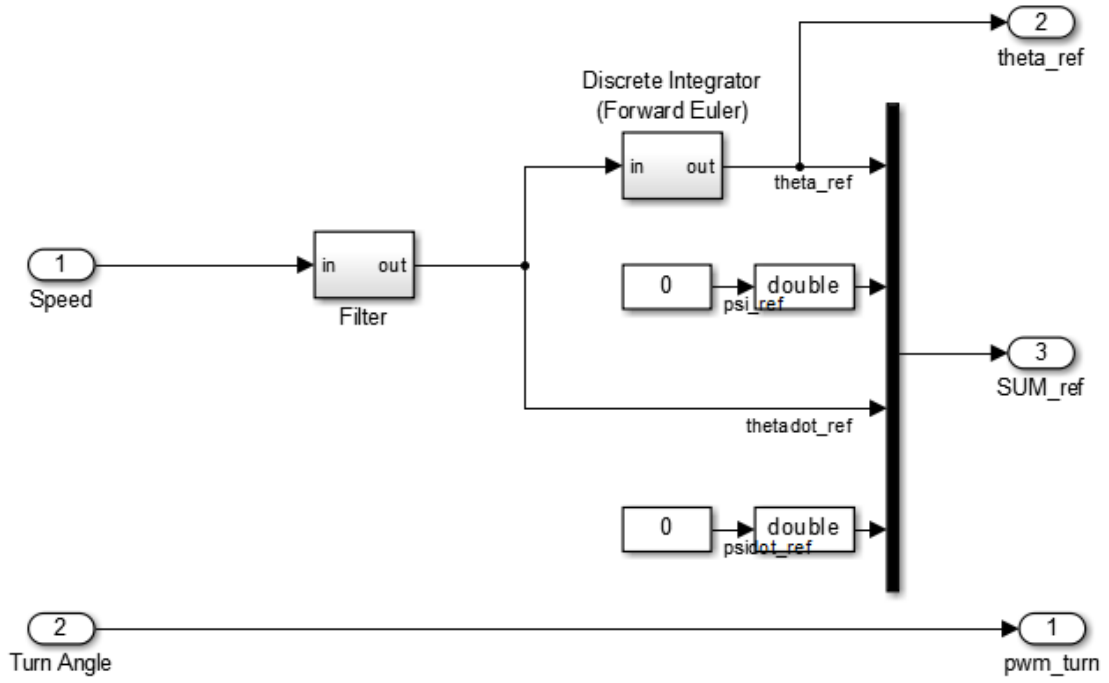

Figure 33 Calibration Reference

The Filter is used to smooth the speed change to make the process of speed change slower, for the stability. The controller is designed as a servo so the only references need to be set by users are Speed and Turn Angle. The Parameters psi_ref and psidot_ref are used to make the robot run in a fix body angle. However, in this project there is no need to do that, so this function can be discussed in the future.

The SUM subsystem is shown below. It is used to calculate the PWM output value for body angle control.

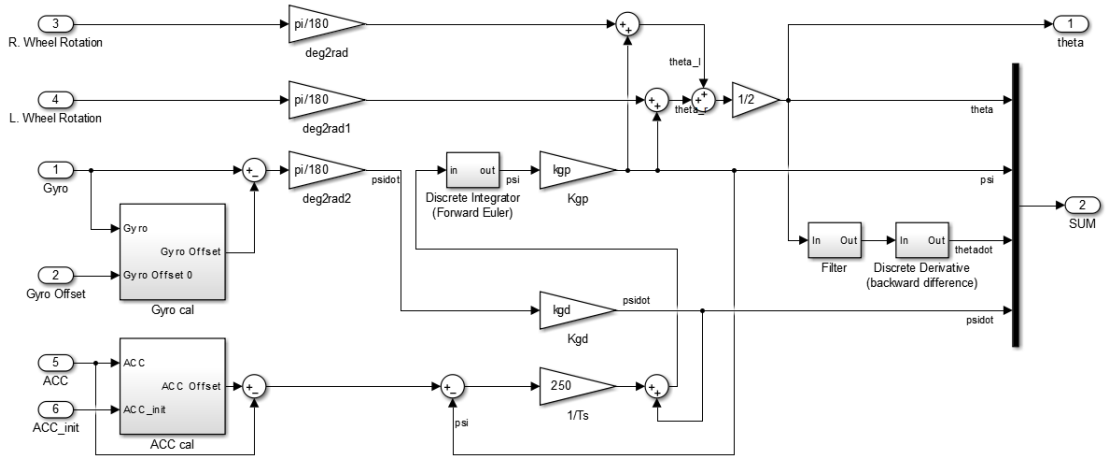

Figure 34 SUM subsystem

This subsystem reads the data from encoders and sensors to calculate the speed value to control the pitch angle. There are also a calibration function used to calibrate the zero point drift of gyroscope and the initial angle setting of accelerometer. The algorithm is discussed in 5.1.3 (1) Angle Control. What's more, the difference of two encoders output is divided by 2 and then added and subtracted to motors. This is for balancing the rotation difference of two motors. In this subsystem, the value of body angle psi, body angular speed psidot, motor rotation theta, motor angular speed thetadot will be calculated and summed together as the part of body angle control. The control method is PD control. The value of the parameter kgp and kgd will be discussed in next section. There is two ways to get those two values, one is calculation in math model, and another is experiments. The best way is to calculate the values in math model then adjust them in experiments.

PWM control subsystem, shown below, is used to get the values of speed and turn angle, and then calculate them into PWM value and send it to motors.

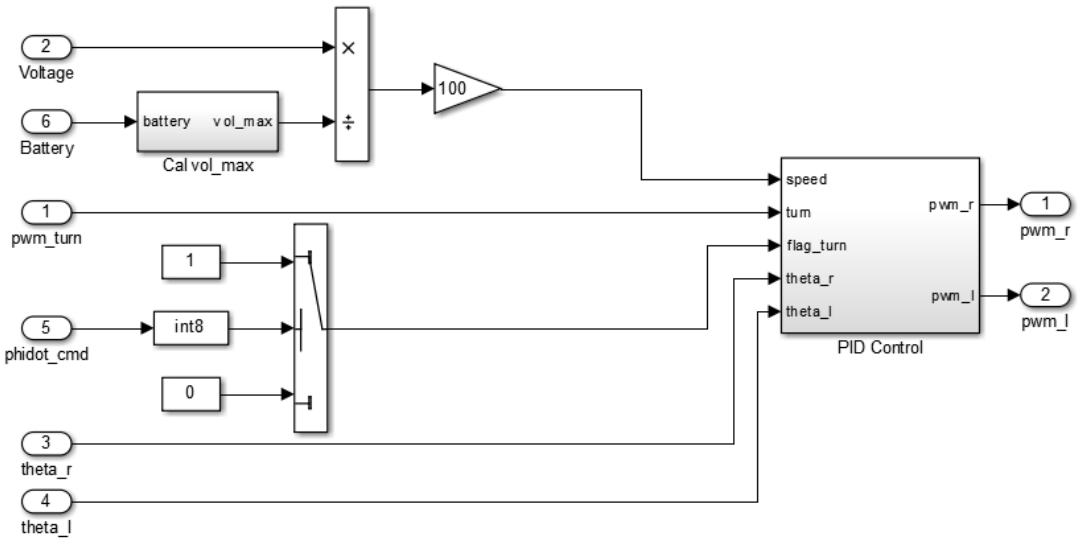

Figure 35 PWM control subsystem

In this subsystem, the PWM value should be noticed. The speed of motor is affected by the battery voltage. For making the robot run a long time, it is necessary to adjust the value of speed to be divided by the battery voltage and then to time it by 100. In this way, the PWM value will be changeable because the battery voltage is decreasing when running, but the real speed will keep same. This method ensures the robot stable even when the battery voltage is low.

Another problem should be noticed is when steering, the robot will be quite unstable because the difference of wheel rotations will be fixed by program, there is not an effective way to fix this problem. As a result, a flag signal should be added to identify if the steering reference is 0. If it is not, means the robot is turning, then the control of speed will be stopped until the steering reference is 0, the detail will be shown later.

The PID control subsystem is shown in Figure 36.

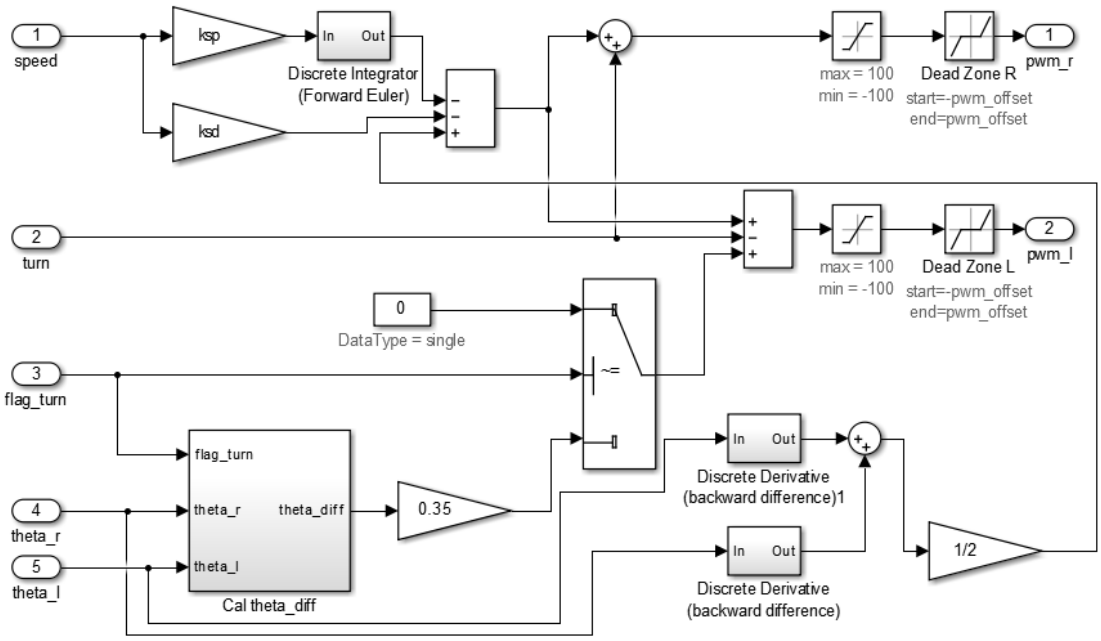

Figure 36 PID control subsystem

The PID control subsystem is part of the PWM control subsystem, used to control the speed and steering of the robot. The speed control part of it is discussed in 5.1.3 (2) Speed Control. Actually the control method here is PD control, the parameter ksp and ksd can be determined by experiments.

The reason why use the PD control is that the PD control is easy to implement and the consumption to microcontroller is low, which is good for improving the performance.

The saturation block is used to set an upper and lower limitation for PWM output, which is from -100 to +100. The dead zone outputs zero for inputs within the dead zone. Offset input signals by either the Start or End value when outside of the dead zone. The dead zone is a method used to decrease the amplitude of robot moving forward and backward when balancing,

After finishing the controller, it can be created as a subsystem, shown below.

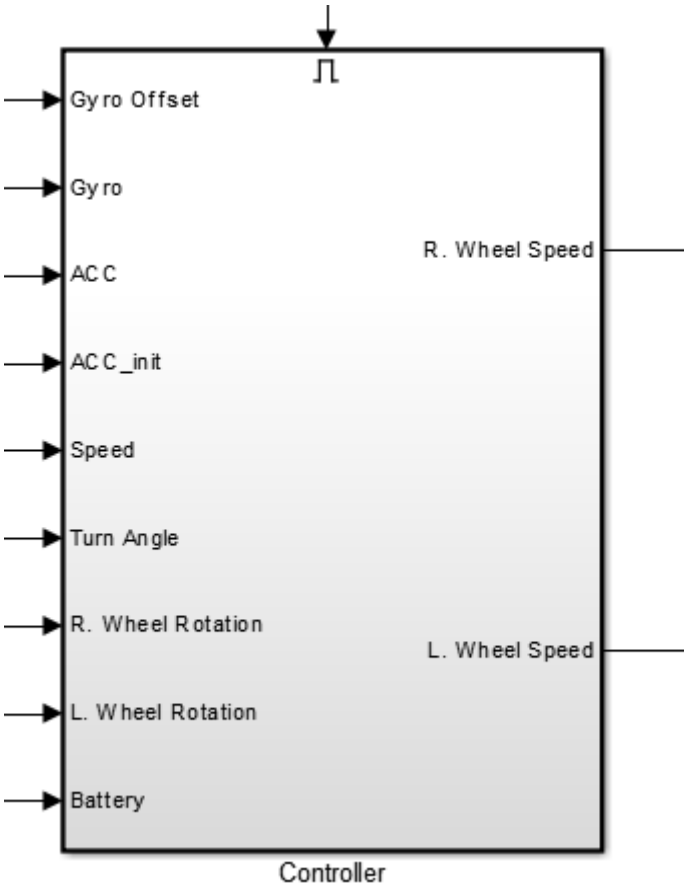

Figure 37 Controller Block

5.2.4 Program overview

After designed the whole program, it can be created as the program below.

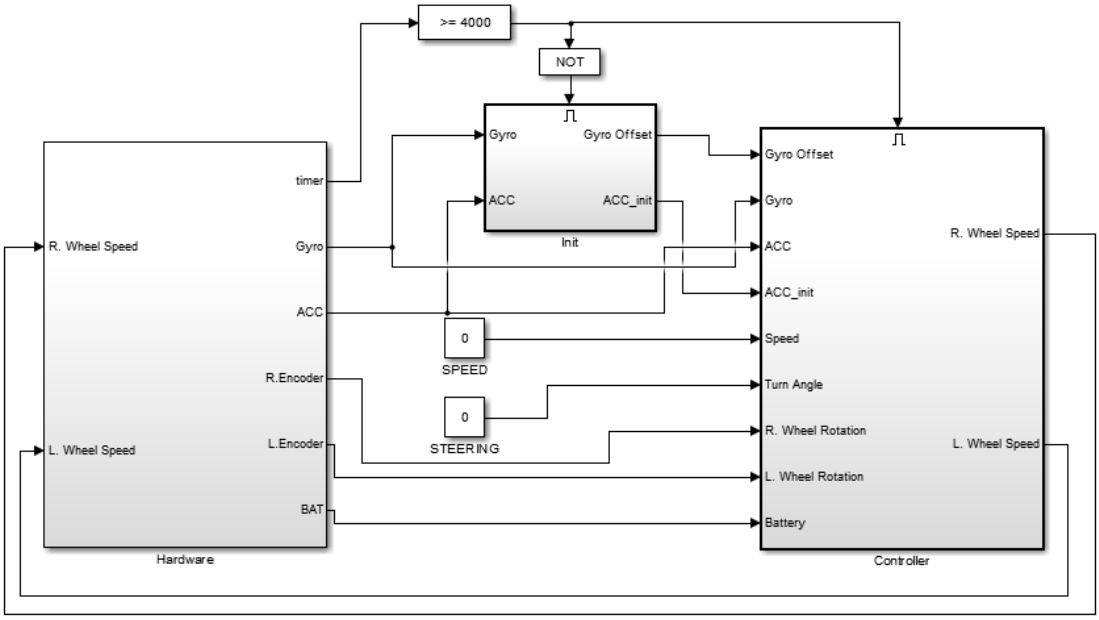

Figure 38 SBTWR self-balance program overview

Basically, it is a double PD closed loop controller. The sensors and encoders send the data to the controller, and then the controller calculate the data and send the speed value to the right and left motors.

As the controller is a servo, the only inputs of this system are speed and steering. In this system, the value of speed should be -5 to +5. The value of steering is the angular speed of robot turning, normally set to 5, meaning 5 degrees per second.

5.3 Parameters setting

There are many parameters in the algorithm program, although these parameters can be optimized and calculated in theory. However, due to the impact of the model accuracy, the parameters obtained can only be used as reference value, the starting range of debugging. The actual optimization parameters need to be determined through a certain engineering step, which is called the parameter setting.

Firstly, there is a need to determine the parameters of the robot. The sensor data parameters is fixed, all of the data is converted to rad value.

The key parameters are motion control parameters, which are slightly different from the theory value because the parameter of each part is not ideal. The exact value of them should be defined in experiments. The testbed and devices are shown below.

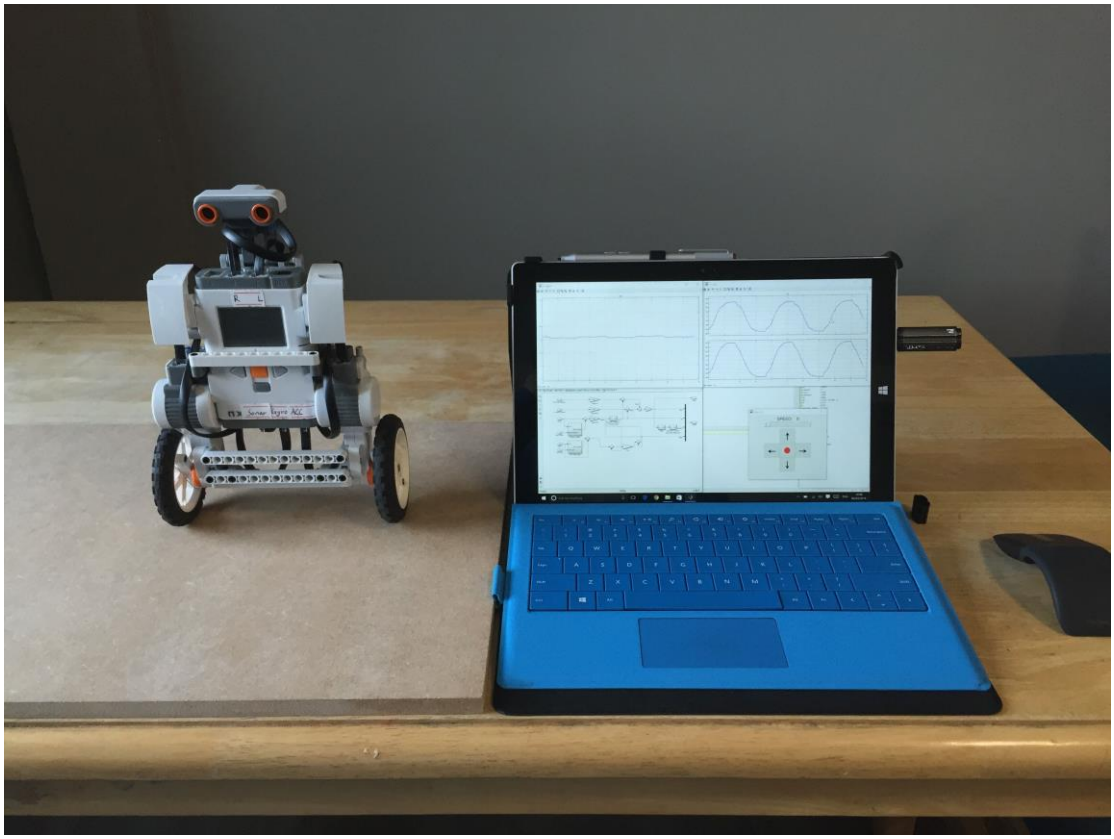

Figure 39 Testbed and devices

To adjust the parameters, hardware in loop simulation is essential. Because the robot is a dynamic system, a cable will affect the movement of it. So it need to be controlled via Bluetooth, the setting is shown below.

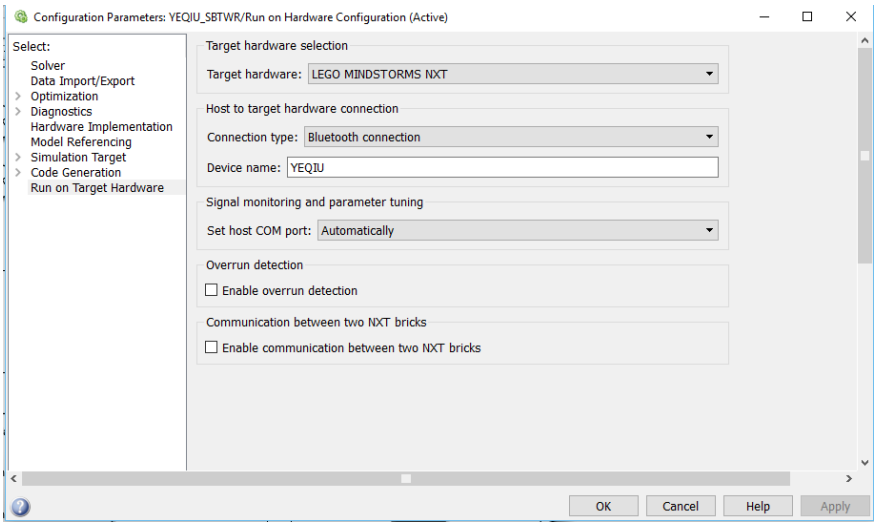

Figure 40 Run on Target Hardware setting

Then set the mode of simulation to External and simulation time is inf, shown below. Click run button and start to adjust the parameters.

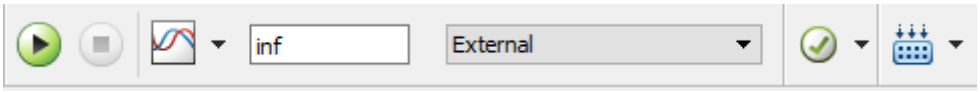

Figure 41 Simulation setting

These parameters include two categories:

(1) Angle control parameters

kgp: Angle proportional control parameter

kgd: Angle derivative control parameters

(2) Speed control parameters

ksp: Speed proportional control parameter

ksd: Speed derivative control parameter

pwm_offset: dead zone parameter

5.3.1 Angle control parameters

The angle control parameters including proportional and derivative parameters two. The proportion parameter (kgp) is equivalent to the restoring force of the inverted pendulum. This parameter needs to be greater than the gravity acceleration effect can make the car upright. With this parameter increases gradually, the robot will begin to be able to stay upright. If this parameter is further increased, the robot will appear swinging back and forth. The derivative parameter (kgd) is equal to the damping force in the inverted pendulum, which can effectively restrain the swing of the robot. If the parameter is too large, it will cause the vibration of the robot itself. This is because the robot is not a rigid body, the body has a certain resonance frequency. If the derivative parameter is too large, it will cause the robot to generate the resonance of the body under the driving of the motor.

Adjustment of these two parameters can follow the process of proportional differential. First, adjust the proportion of parameters, so that models can be kept upright and began to swing back and forth. It shows that the proportion of the matter has been able to overcome the gravity of the image. And then gradually increase the derivative parameters, making the robot to gradually maintain a stable. Increase the derivative parameters until the robot begins to resonate, so that the maximum value of the derivative parameters can be determined. Then appropriate to reduce the derivative parameters, and then gradually increase the proportion of parameters, until the robot began to tremble, so that the maximum value of the parameters can be determined. And then appropriate to reduce the proportion of parameters. In the vicinity of these parameters for the test until find the optimal control of the optimal parameters.

Finally, for the SBTWR in this project, the angle control parameters are determined, kgp = 0.25 and kgd = 0.003.

5.3.2 Speed control parameters

After the initial determination of the angle control parameters, the setting of the speed control parameters can be carried out. The speed control parameters include the proportion (ksp) and the derivative (ksd) control parameters, the proportion and the differential definition are as to the definition of feedback speed.

Firstly, the proportion of control parameters determine the ability to inhibit the speed error, but if there is only the proportional control, robot speed control will appear double integral negative feedback. The first integral is the integral of the error, and the second integral is the acceleration integral produced by the inclination of the robot. So the speed will tremble. The derivative control can eliminate this oscillation phenomenon.

Then in the adjustment, because it is carried on the static adjustment, speed control closed loop speed setting value can be set to 0. After that, start the adjustment of the ratio and the differential parameters. Specific debugging process can be the first step in the proportion parameter, robot will move back and forth around a certain balance point. Then gradually increase the derivative parameter value, robot will soon be stuck in a balance point out. Using the external force impact models, models can quickly become stationary. Then gradually increase the proportion and differential control parameters, making the car models to resist external interference and the impact of the ability to gradually increase.

As a result, for the SBTWR in this project, the speed control parameters are determined, ksp = 0.02 and ksd = 0.125.

In the robot power transmission system, each part has the static friction, it will reduce the static stability of the robot. It can be overcome by adding a dead zone constant to the motor control voltage.

When the dead zone constant is 0, the model will appear in the static state back and forth movement of small amplitude. By gradually increasing the dead zone constant, the moving phenomenon of the robot will gradually disappear. When the dead time constant is too large, there will be the phenomenon of motor vibration.

However, in real use, the robot is stable enough, so the dead zone constant can be set to 0 as default.

6. Additional Features

6.1 Bluetooth control

In External Simulation Mode, the robot can be controlled via Bluetooth by changing the speed and turn angle constants. A GUI is designed to implement it, shown below.

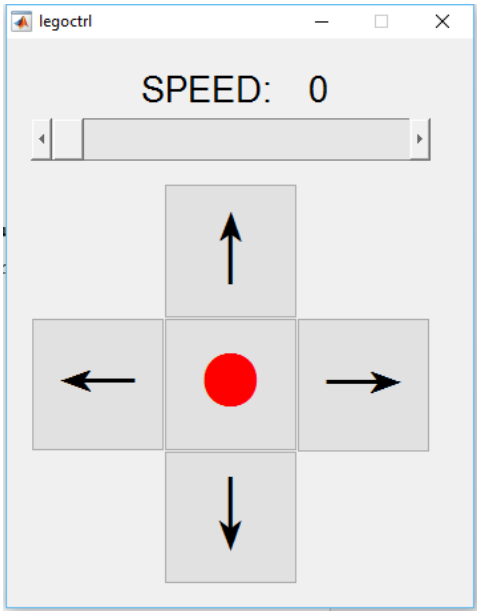

Figure 42 Control panel for LEGO SBTWR

By setting the speed and click Up or Down buttons can make the robot move forward or backward. Right and Left buttons are used to turn the robot, click once will make the robot turn by about 5 degrees. The red button in the centre is emergency stop button. This is also used in the parameter setting, to stop the robot when it is out of control. The GUI design code is shown in next page.

```
% --- UP
function pushbutton1_Callback(h0bject, ~, handles)
set_param('lego_selfbalance/Constant1','Value','slider1.value');
guidata(h0bject, handles);

% --- DOWN
function pushbutton2_Callback(h0bject, ~, handles)
set_param('lego_selfbalance/Constant1','Value','-1*slider1.value');
guidata(h0bject, handles);

% --- RIGHT
function pushbutton3_Callback(h0bject, ~, handles)
set_param('lego_selfbalance/Constant2','Value','20');
pause(0.1);
set_param('lego_selfbalance/Constant2','Value','0');
guidata(h0bject, handles);


% --- LEFT
function pushbutton4_Callback(h0bject, ~, handles)
set_param('lego_selfbalance/Constant2','Value','-20');
pause(0.1);
set_param('lego_selfbalance/Constant2','Value','0');
guidata(h0bject, handles);

% --- Emergency stop
function pushbutton5_Callback(h0bject, ~, handles)
set_param('lego_selfbalance/Constant2','Value','0');
set_param('lego_selfbalance/Constant1','Value','0');
set_param('lego_selfbalance/Gain1','Gain','0');
set(handles.text5,'String','0');
set(handles.slider1,'Value',0);
guidata(h0bject, handles);

% --- Speed setting
function slider1_Callback(h0bject, ~, handles)
speed = get(h0bject,'Value');
set_param('lego_selfbalance/Gain1','Gain',num2str(speed));
set(handles.text5,'String',num2str(speed));
guidata(h0bject, handles);
```

6.2 Obstacle detection

The ultrasonic sensor in LEGO NXT set can be used to detect the obstacle in front of the robot. It outputs the distance between obstacles and itself in centimetres from 0 - 255. The basic idea is when the obstacle in front of robot appears within 20cm, the speed of robot will be forced to set to 0. The graphic program is shown below.

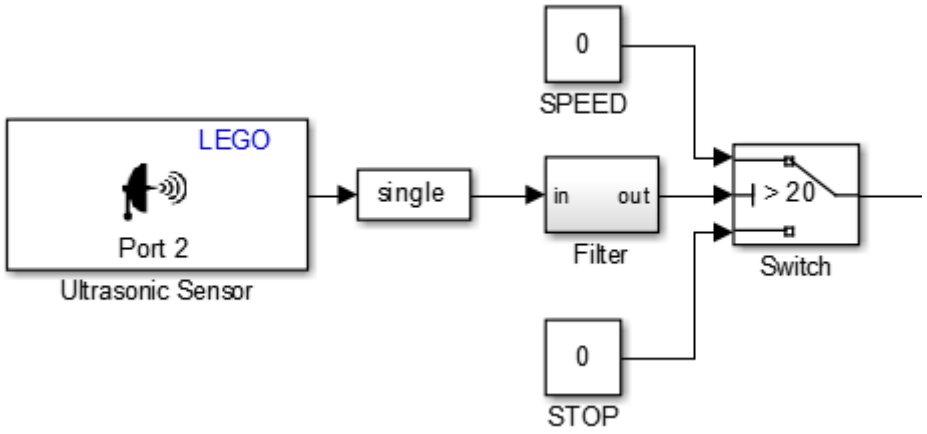

Figure 43 Obstacle detection

The low pass filter is used to smooth the distance changes. As described in 4.1, the ultrasonic sensor is set on the top of the robot.

By now this function is quite easy, which is just based on a comparator. However, it can be optimized in the future by using PID control method to maintain the distance between obstacles and robot.

6.3 Multifunctional mobile platform

This sub-section is shown the joint work with Tao Zhang, whose project is "Ultrasonic Indoor Localization System for Unmanned Vehicles/Robots".

The SBTWR in this project is used as a mobile platform to help Tao test the localization system.

The experiment is shown below.

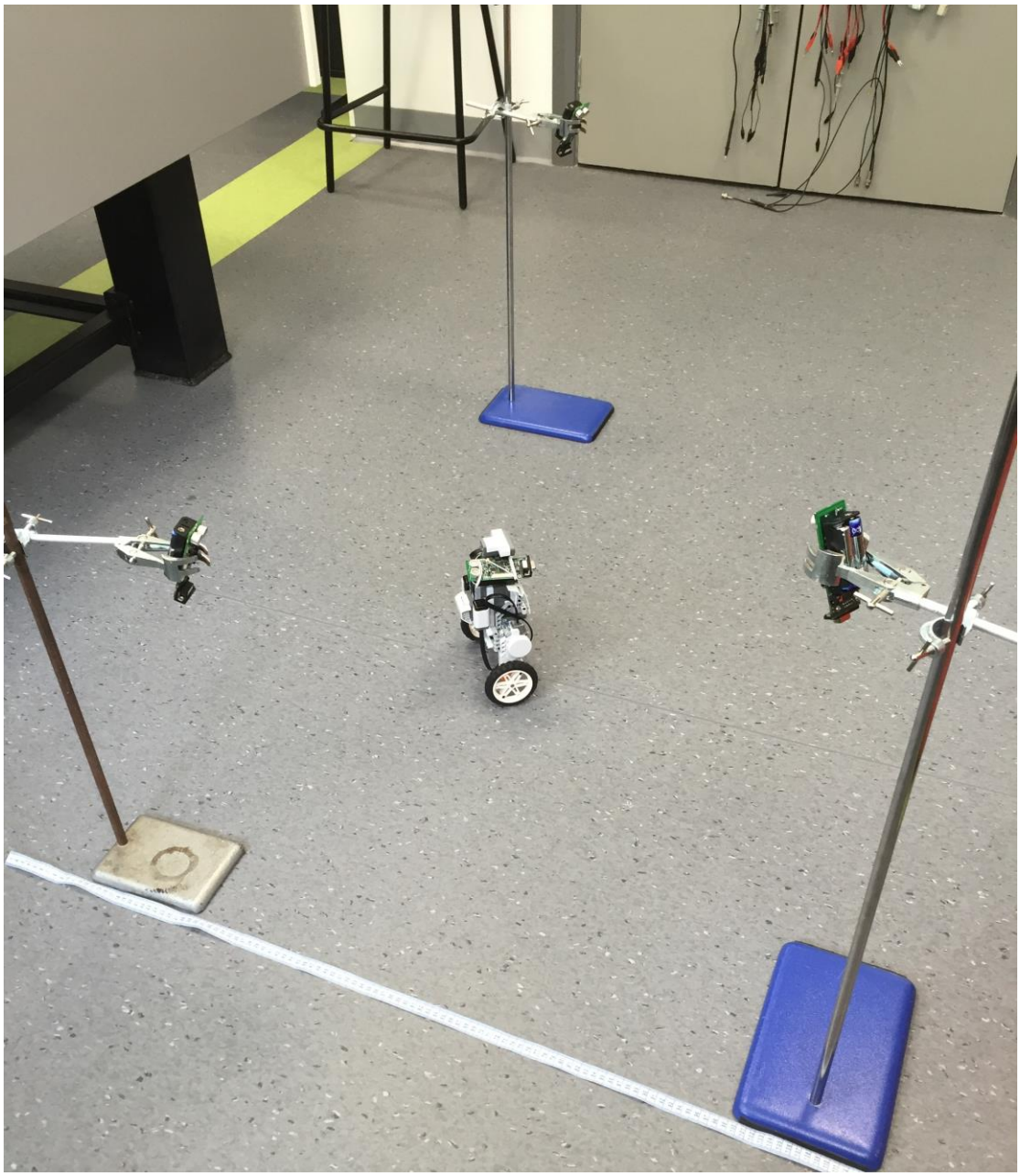

Figure 44 the Experiment of Indoor Localization System

The detail picture is shown below. Because the listener of that system has independent power source, so basically it is just fixed on the shoulder of the robot.

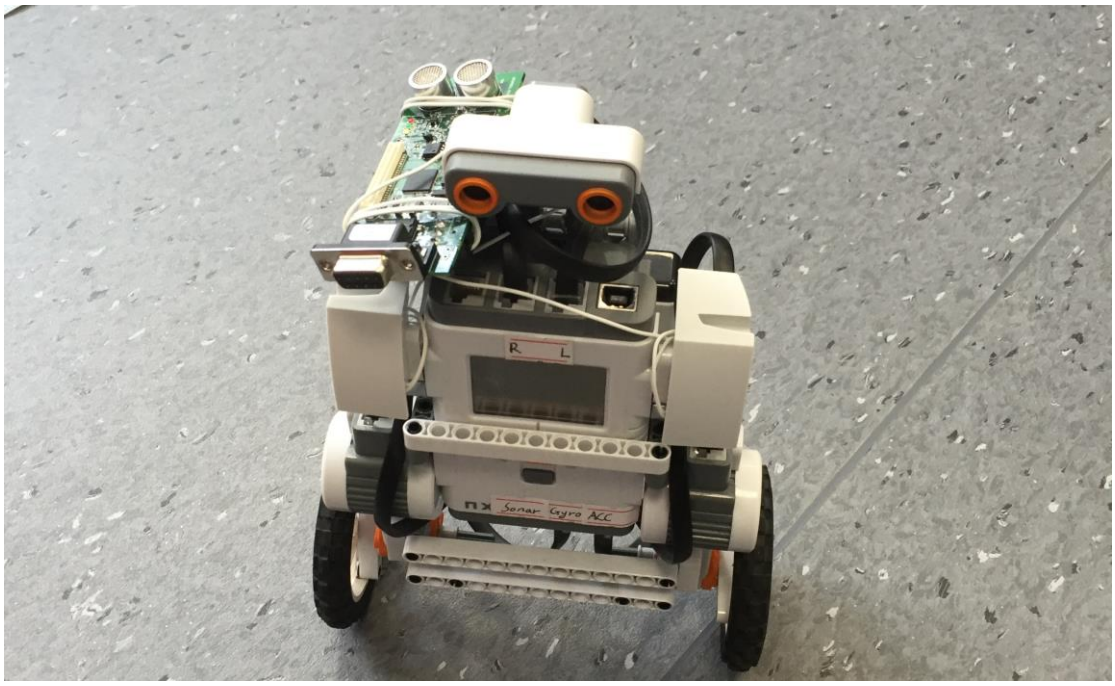

Figure 45 SBTWR with Localization Listener

This experiment also shows the ability of the SBTWR in this project as a mobile platform. It can also be used in any other experiments which need mobile platform.

7. Test and Results

7.1 Stability test in simulation

By setting up a simulated model in Matlab, a simulated robot can be constructed. The mathematical model has been derived in 2.2. The Simulink model is supported in the support package, shown below. Because the simulated gyroscope is ideal, there is no need to add an accelerometer to offset the drift of gyroscope.

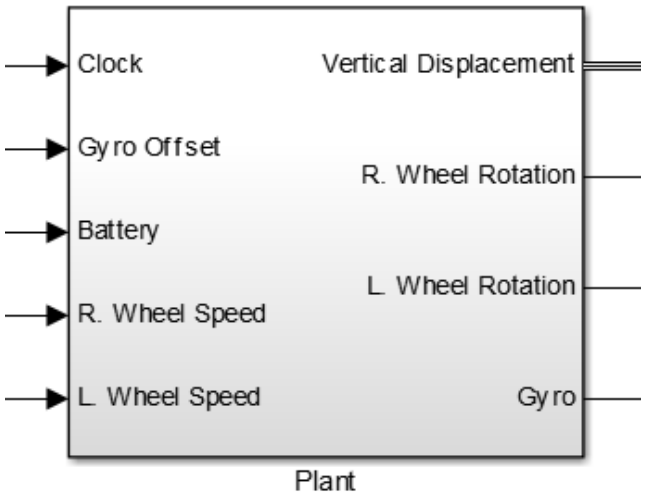

Figure 46 Simulated plant in support package.

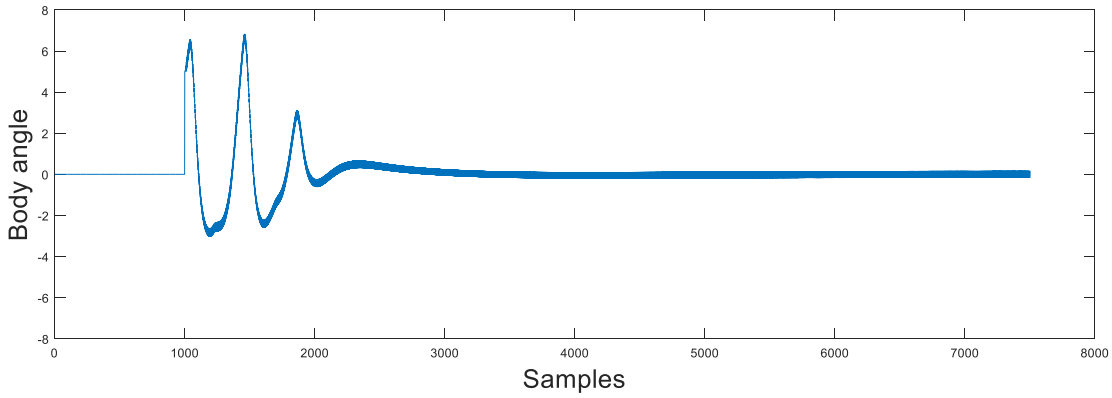

Figure 47 the Body Angle in simulation

The Figure 45 is shown the body angle changes in for 30 seconds. The simulation is carried out in 30s, the frequency of the conduction is 250Hz. So there are 7500 samples. This result shows the robot can be stable in 4 around seconds (from sample 1000 to 2000). The initial speed is set to 5 so the integration of positive angle and negative angle should be bigger than 0. This figure is shown clearly how the robot stay in balance from an unstable state.

7.2 Actual robot stability test

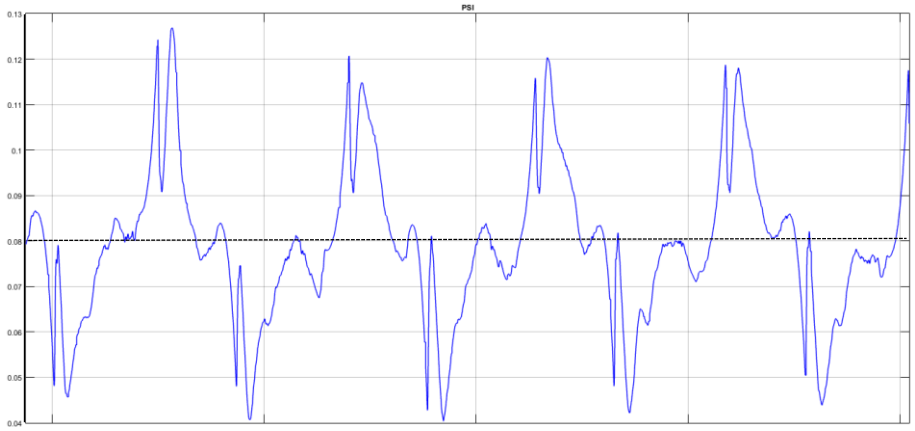

Figure 48 the Body Angle from real-time data feedback

The data of body angle can be send back to computer via Bluetooth in real-time. The body angle in real is from -0.08 rad to +0.10 rad, shown in Figure 46. It means the robot is swinging from -4.58 degrees to +5.73 degrees. This result is shown the robot body angle changes in balancing state, and it is always changeable. In the other words, the robot can keep a dynamic stable state in real, shown in Figure 49.

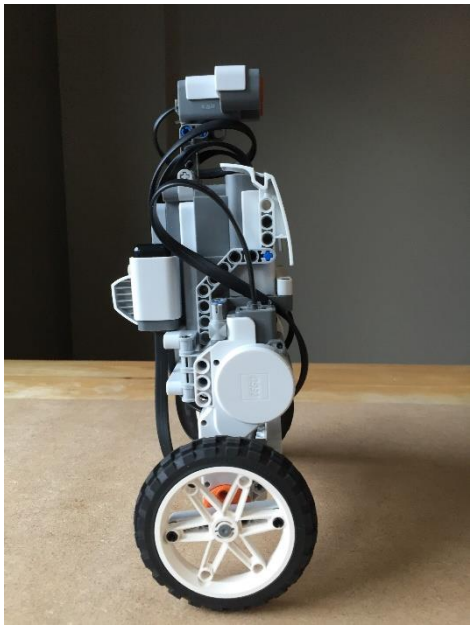

Figure 49 the SBTWR in stable state

7.3 Slope climbing ability test

This test is to roughly check the robustness and stability of the SBTWR in this project. The main idea is to control the robot to climb the slopes with different inclinations. As shown below, the robot is quite stable on the slopes.

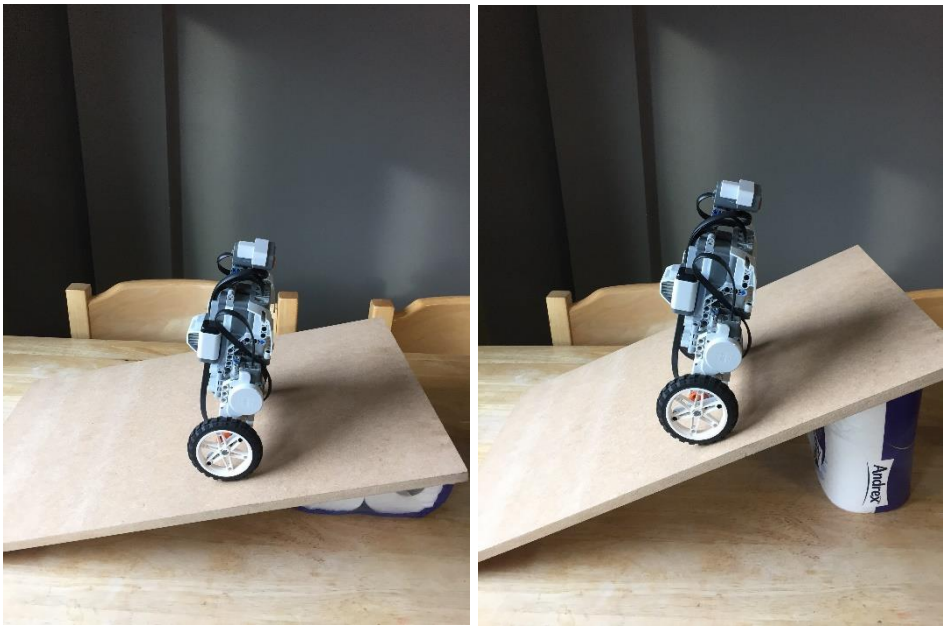

Figure 50 Climbing capacity test

Moreover, the reason why the robot has such a good robustness is that the parameters are tested and adjusted many times in experiments. Also, the big wheels with sporty tyres supply a bigger friction between tyres and surface of slopes, which prevent the slipping.

8. Discussion

8.1 Work have done

Based on the combination of analytic mechanics and classical mechanics, the mathematical model of the system are built. What's more, according to the established model, the basic physical characteristics are discussed, the relevant conclusions are reached. Moreover, to the controller design of body angle and the wheel speed, the optimized double PID closed-loop controller with better control effect is built, and the robustness of which is verified by simulation experiment.

As the SBTWR is constructed by using LEGO MINDSTORMS NXT set, the Model-Based Design technique is used to develop the control algorithm for it. Furthermore, the control algorithm is converted to program code by code generator and written into the control chip. Then after a series of experiments are taken, the results prove the controller designed in this report effective for real system too.

8.2 Critical appraisal

This report uses the theoretical knowledge as little as possible to introduce the control method of self-balancing two-wheel robot. For example, the best method to control the body angle is LQR (linear quadratic regulator), but in this project I used PID control because it is easier to understand and implement. However, it still increased my practical experience in hardware and software development. The problems and difficulties I faced also help me to improve my ability.

What's more, this report can be used as a reference to develop a SBTWR, the methods such as PID control and parameters setting are easy to learn, the technologies in this report are all explained.

9. Conclusion

This project provided a great chance to study control theory and embedded system. By overcoming different problems in the development of this project, I learned how to face the difficulties and challenges as an engineering student. Also, this project provided the experience of developing an dynamic system, which enhanced my ability and it is significant to an engineering student in the future.

9.1 Achievements

After the construction, programming, debugging, running in the real environment, to find the existence of errors, optimize the various control parameters, and gradually improve the robustness of the SBTWR, a set of optimal control strategy is obtained by continuous researching and exploring.

9.1 Further work

In the running test, the control algorithm can be further improved. To increase the state view of the SBTWR measurement, the nonlinear part is modified, and the stability and anti-disturbance ability is improved. Based on the test results have been collected, gradually improve the control method, break through the limitation of current ideas, and keep seeking more advanced and great control methods, to achieve a further knowledge about control theory.

Reference

[1] MathWorks, "Self-Balancing Two-Wheel Robot", MathWorks, [Online]. Available: http://uk.mathworks.com/help/supportpkg/legomindstormsnxt/examples/self-balancing-two-wheel-robot.html. [Accessed 12/10/2015]

[2] Wikipedia, "Inverted pendulum", Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Inverted_pendulum. [Accessed 15/102015]

[3] Felix Grasser, Aldo D'Arrigo, Silvio Colombi and Alfred C. Rufer, "JOE: A Mobile, Inverted Pendulum", IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, VOL. 49, NO. 1, February 2002, p.108

[4] Wikipedia, "PID controller", Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/PID_controller. [Accessed 16/10/2015]

[5] Yorihisa. Yamamoto, "NXTway-GS Model-Based Design", MathWorks, January 2009, p.1.

[6] Jack Little, "The Impact of Model-Based Design on Controls," MathWorks, August 2014, p.57.

[7] Philo, "Get up, NXTway!", Philo's Home Page, [Online]. Available: http://www.philohome.com/nxtway/nxtway.htm. [Accessed 30/10/2015]

[8] LEGO MINDSTORMS NXT, "LEGO MINDSTORMS NXT Hardware Developer Kit", LEGO, p.3.

Bibliography

1. W. Yao, "Research on Motion Control Technology of Two-wheeled Selfbalancing Electrical Vehicle", CNKI, 2010.

2. Y. Yamamoto, "NXTway-GS Building Instructions", Mathworks, 2008.

3. Agam Kumar Tyagi, "MATLAB and SIMULINK for Engineers (Oxford Higher Education)", OUP India, Pap/Cdr edition (8 Dec. 2011)

4. Harold Klee, "Simulation of Dynamic Systems with MATLAB and Simulink, Second Edition", CRC Press, 2 edition (25 Feb. 2011)

5. Dr Xuewu Dai, "Rapid BeagleBoard Prototyping with MATLAB and Simulink", Packt Publishing (25 Oct. 2013)

6. W. Kenneth Jenkins, "Advanced Concepts in Adaptive Signal Processing", Kluwer Academic Publishers (30 Jun. 1996)

Appendix 1 Pictures of the SBTWR in this project

Figure 51 Front view of the SBTWR

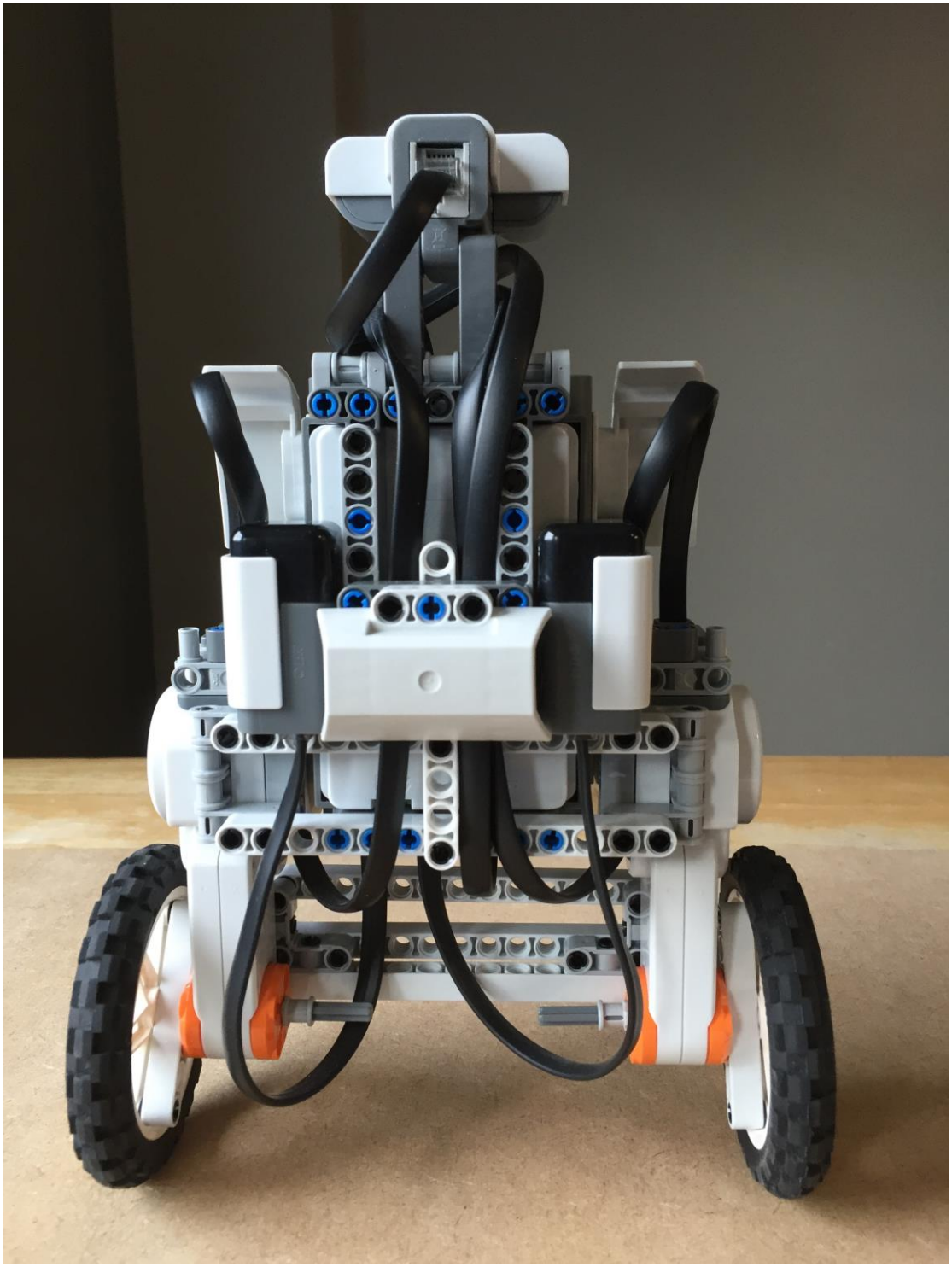

Figure 52 Back view of the SBTWR

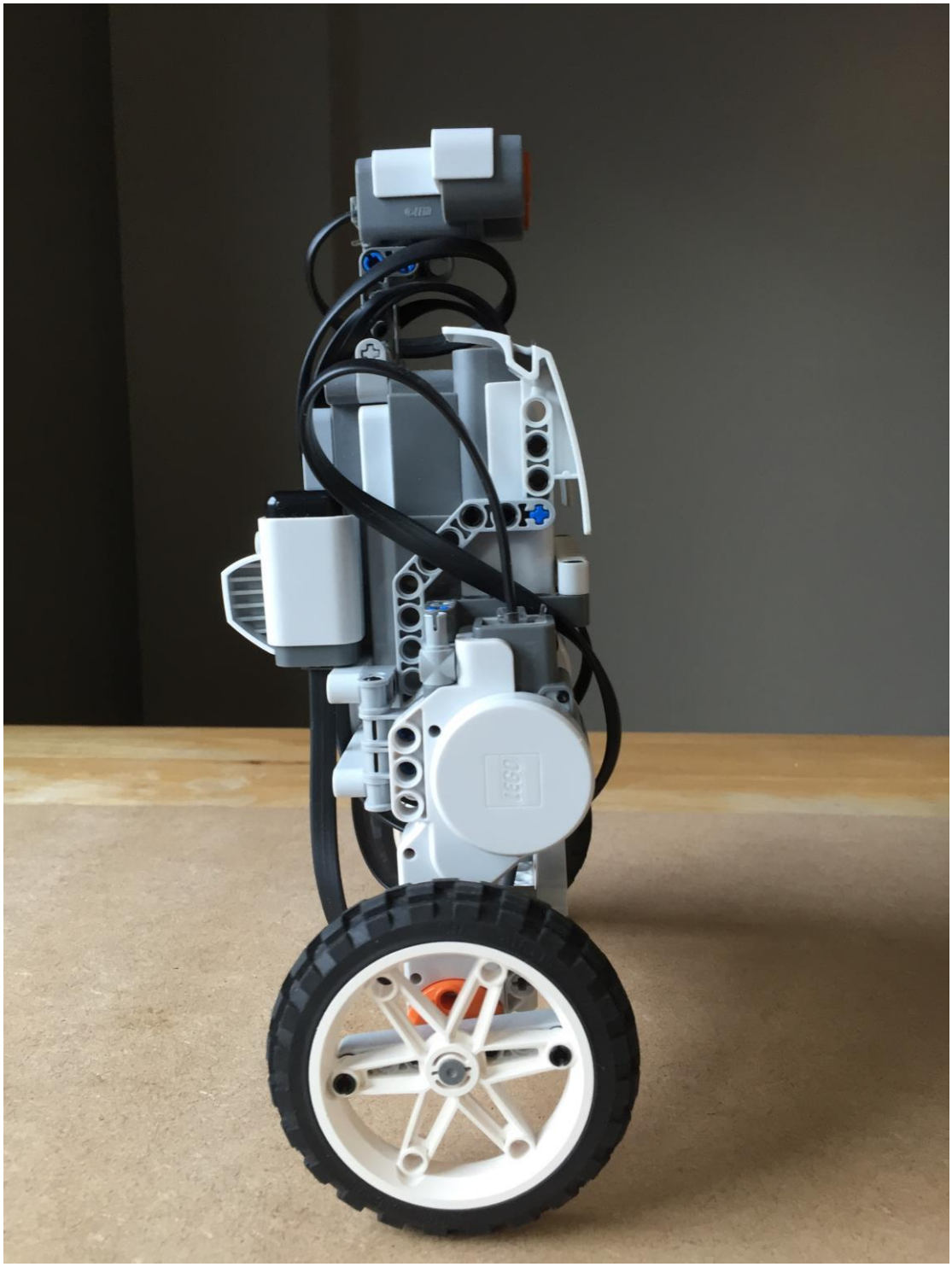

Figure 53 Side view of the SBTWR

Appendix 2 Project management

Ⅰ Time-scales

a. Proposed Schedule and Timeline (Gantt chart)

Figure 54 Gantt chart and Timeline

b. Final time-scales

Figure 55 Final time-scales

Ⅱ Resources and Components

a. Software

Software	Release	Supplier
MATLAB	R2015a	Northumbria University
Simulink	R2015a	Northumbria University
LEGO MINDSTORMS NXT Support from Simulink	R2015a	http://uk.mathworks.com/

Table 2 Software

The support package for LEGO® Mindstorms® NXT is a set of modularized block in Simulink Library, used to develop graphic program. The blocks are shown in Figure below.

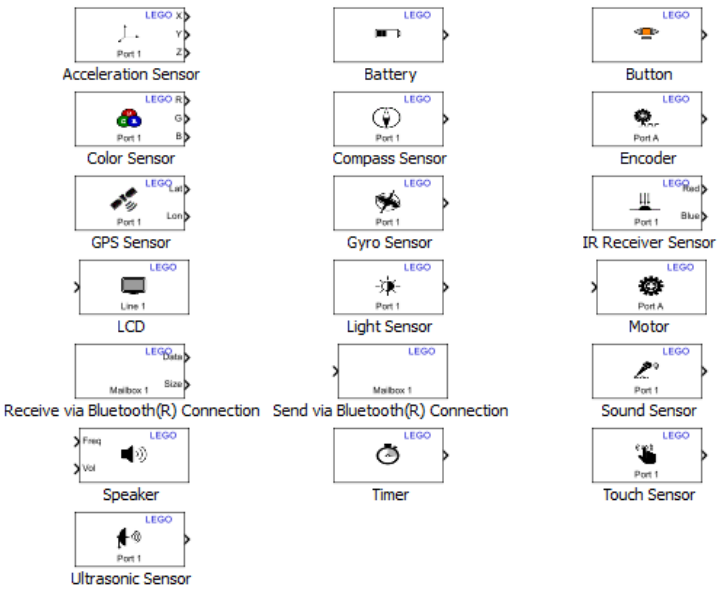

Figure 56 The Simulink Blocks of support package for LEGO Mindstorms NXT

With the support of blocks package, the developer don't need to spend extra time to study the underlying drivers of all input/output ports and relative components, what supplies a great convenience to developers.

b. Hardware

Product	Product ID	Supplier
LEGO Mindstorms NXT 2.0	8547	Northumbria University
Lego Mindstorms NXT Resource Set	9695	http://www.amazon.co.uk/Lego
HiTechnic NXT Gyro Sensor	NGY1044	http://www.hitechnic.com/
NXT Acceleration / Tilt Sensor	NAC1040	http://www.hitechnic.com/

Table 3 Hardware

Lego Mindstorms NXT Resource Set shown in Figure 57 is necessary for developing the SBTWR because it supplies some parts excluded in the LEGO® Mindstorms® NXT 2.0 set. The gyroscope and accelerometer (they are packaged in same plastic shell) used in this project, shown in Figure 58, are from a third party company called HiTechnic. HiTechnic manufactures a range of robotic sensors for the LEGO® MINDSTORMS® NXT. The majority of HiTechnic sensors are certified by The LEGO Company assuring the highest standards of quality and safety.

Figure 57 Lego Mindstorms NXT Resource Set

Figure 58 HiTechnic gyroscope and accelerometer

III Costing analysis

a. The expected capital costs

Software	Release	Supplier	Price
MATLAB	R2015a	Northumbria University	Free
Simulink	R2015a	Northumbria University	Free
Simulink support package for Lego Mindstorms NXT	R2015a	http://uk.mathworks.com/	Free

Table 4 Software cost

Northumbria University supplies the student license of MATLAB, so the latest version of MATLAB can be accessed for free. Also, the support package can be downloaded in Simulink for free.

Product	Product ID	Supplier	Price
LEGO Mindstorms NXT 2.0	8547	Northumbria University	Free
Lego Mindstorms NXT Resource Set	9695	http://www.amazon.co.uk/Lego	£79.44
HiTechnic NXT Gyro Sensor	NGY1044	http://www.hitechnic.com/	£35.59
NXT Acceleration / Tilt Sensor	NAC1040	http://www.hitechnic.com/	£35.59

Table 5 Hardware cost

Northumbria University supplies the LEGO® Mindstorms® NXT 2.0 set for free.

According to Table 4, the cost of hardware is:

$$£79.44 + £35.59 + £35.59 = £150.62$$

b. The cost of utilisation of existing resources

The existing software and hardware resources is same as that in proposal. As a result the cost of existing resources is £150.62.

c. The manpower cost

Estimate the manpower costs expanded on the project assuming a salary of £20 per hour.

Project Development	65 days	Tue 03/11/15	Sun 31/01/16
Final Report	53 days	Mon 01/02/16	Wed 13/04/16

Table 6 Main tasks time-scales

According to the time-scales above. The duration of development is 65 + 53 = 118 days. Take 3.5 hours as average working hours per day, the number of hours required to develop should be

$$118 \times 3.5 = 413$$

Thus the cost of manpower is

$$413 \times £20 = £8260$$

As a result, the total costing of development can be calculated, which is

$$£150.62 + £8260 = £8410.62$$

Ⅳ Comparison

a. Comparison between current work and proposal

The work done covers the aims and objectives in proposal. Although some implementation methods are not exactly same as that in proposal, the functions and characteristics meet the requirement.

b. Comparison between time-scales

Generally, the work is carried out smoothly as the Gantt chart in proposal. Some sub-tasks costed more time than that in proposal because of various reasons such as the out of stock of sensors or some parts, but the main tasks is completed in the scheduled time.

c. Comparison between costs

The expected capital costs and cost of utilisation of existing resources are same because the hardware components in this project are not changed.

Appendix 3 ETHICS REGISTRATION AND APPROVAL FORM

Section One: Registration *[To be completed by researcher]*

Title of research project/dissertation	**Control and Self-Balancing of Two-wheel Robots**

Researcher's name	**YEQIU TANG**

Please only complete the following if researcher is a student:

Programme of study	**Electrical and Electronic Engineering**

Academic Year	**2015/2016**

Module code (if applicable)	**EN0624**

Principal Supervisor or Module Tutor	**Ian Forbes**

Start Date	**23/09/2015**

Brief outline of research topic:

This project is to design a controller for a self-balancing two-wheel robot, which can keep the robot in vertical balance and control it to move around on the ground.

Short description of proposed research methods including identification of participants:

The main methods used in this project are mathematic modelling technique and PID control. The modelling method is used to make an math model for the two-wheeled inverted pendulum to calculate the parameters of the controller. The PID control is used to maintain the vetical balance and control the speed for the robot.

Ethical considerations in the research project	YES	NO
1. Does your research involve an external organisation or partner?	☐	☐
2. Does your research involve human participants?	☐	☐
3. If yes to Q.2, will you inform the participants about the research?	☐	☐
4. Will you obtain their consent using the standard consent form?	☐	☐
5. Is any deception involved?	☐	☐
6. Do any participants constitute a 'vulnerable group'? (refer to definition of Vulnerable People)	☐	☐
7. Will the research involve the following information? Commercially sensitive Personally sensitive Politically sensitive Legally sensitive	 ☐ ☐ ☐ ☐	 ☐ ☐ ☐ ☐
8. Is the research likely to have any significant environmental impacts?	☐	☐
9. Are there likely to be any risks for the participants in your research?	☐	☐
10. Are there likely to be any risks for you in conducting the research?	☐	☐
11. If yes [to 5, 6, 7, 8, 9 or 10 above] have you identified steps to address the issues and mitigate any risks to participants, yourself or the environment?	☐	☐

Statement to explain how any issues identified above will be addressed and what steps will be taken to mitigate such risks or adverse impacts

Ethical category of research project

Based on the above Ethical Considerations and with reference to the University's Ethical Scrutiny Risk Assessment tool, identify the Ethical category of your research project (refer to http://www.northumbria.ac.uk/static/5007/respdf/riskassesmenttool for further guidance):

[Please tick as appropriate]

Red	☐	vulnerable participants; human tissue; sensitive data; risks to participants & researchers etc.
Amber	☐	human participants requiring informed consent; commercially sensitive information etc.
Green	☐	no participants involved; secondary data only; no sensitive data

I have read the University and the Faculty Ethics Policy and Procedures and confirm that the answers I have given above are correct. Where issues arise under items 5, 6, 7, 8, 9 or 10 [above] I have described in writing how I intend to approach these issues in the research.

Researcher's signature .....

Date .....

Section 1 Ethics Registration to be submitted to Principal Supervisor or Module Tutor and allocated to a reviewer as follows:

Green risk - may be approved by Supervisor
Amber risk - to be submitted for approval by one independent reviewer
Red risk - to be submitted for approval by two independent members of Faculty Research Ethics Committee

Section Two: Approval

Supervisor/Module Tutor's name confirming ethical risk status	

Ethical approval *[Please tick as appropriate]*

Green - Ethical approval is given without conditions (supervisor may approve)	☐
Amber – (to be approved by one independent reviewer) Ethical approval is given with the following conditions: Information to be provided to all participants Participant consent to be obtained using the standard Research Participant Consent Form or otherwise in accordance with Faculty procedures Data to be stored and destroyed securely in accordance with University guidelines Adherence to Data Protection Act Anonymity to be offered to participants Commercial confidentiality to be provided to organisations(s) Other (please state):	 ☐ ☐ ☐ ☐ ☐ ☐ ☐
Red - Project is referred to FREC for approval.by two independent reviewers *Please e-mail the submission to PGR Faculty Support*	☐

Name & role of reviewer 1:	……
Signature	……
Date	……

Name & role of reviewer 2:	……
Signature	……
Date	……

Outcome of Review